Activity Recognition via User-Trace Segmentation

JIE YIN
CSIRO ICT Centre
QIANG YANG
Hong Kong University of Science and Technology
DOU SHEN
Microsoft Adcenter Labs
and
ZE-NIAN LI
Simon Fraser University

A major issue of activity recognition in sensor networks is automatically recognizing a user's high-level goals accurately from low-level sensor data. Traditionally, solutions to this problem involve the use of a location-based sensor model that predicts the physical locations of a user from the sensor data. This sensor model is often trained offline, incurring a large amount of calibration effort. In this article, we address the problem using a goal-based segmentation approach, in which we automatically segment the low-level user traces that are obtained cheaply by collecting the signal sequences as a user moves in wireless environments. From the traces we discover primitive signal segments that can be used for building a probabilistic activity model to recognize goals directly. A major advantage of our algorithm is that it can reduce a significant amount of human effort in calibrating the sensor data while still achieving comparable recognition accuracy. We present our theoretical framework for activity recognition, and demonstrate the effectiveness of our new approach using the data collected in an indoor wireless environment.

We would like to thank the Hong Kong Research Grants Council (RGC) for its support with Grant HKUST 6187/04E. Part of this work was also carried out in the Tasmanian ICT Centre, which is jointly funded by the Australian Government through the Intelligent Island Program and the CSIRO. The Intelligent Island Program is administered by the Tasmanian Department of Economic Development.

Authors' addresses: J. Yin, CSIRO ICT Centre, Locked Bag 17, North Ryde, NSW 2113, Australia; email: jie.yin@csiro.au; Q. Yang, Department of Computer Science and Engineering, Hong Kong University of Science and Technology, Clear Water Bay, Kowloon, Hong Kong, email: qyang@cse.ust.hk; D. Shen, Microsoft Adcenter Labs, One Microsoft Way, Redmond, WA 98052; email: doushen@microsoft.com; Z.-N. Li, School of Computer Science, Simon Fraser University, Burnaby B.C., Canada V5A 1S6; email: li@cs.sfu.ca.

Categories and Subject Descriptors: I.2.m [**Artificial Intelligence**]: Miscellaneous; H.1 [**Models and Principles**]

General Terms: Algorithms, Performance

Additional Key Words and Phrases: Activity recognition, segmentation, motion patterns

ACM Reference Format:

Yin, J., Yang, Q., Shen, D., and Li, Z.-N. 2008. Activity recognition via user-trace segmentation. ACM Trans. Sens. Netw., 4, 4, Article 19 (August 2008), 34 pages. DOI = 10.1145/1387663.1387665 http://doi.acm.org/10.1145/1387663.1387665

1. INTRODUCTION

A major objective of artificial intelligence and pervasive computing is recognizing high-level goals and activities from low-level signals obtained through sensors. With recent advances in pervasive computing technology, it is now possible to track a mobile user's context information as streams of sensor data. Being able to accomplish activity recognition plays an important role in the development of many important applications. A typical example is to help individuals with cognitive disorders live safely and independently at home [Patterson et al. 2003]. Other potential applications include the provision of timely and useful context-dependent services for shoppers and logistic support. In supporting activity recognition, a challenge is to enable activity recognition through low-cost and universally available wireless sensor networks and wireless local area networks. In this article, we address the problem of recognizing a user's high-level activities from low-level radio-frequency (RF) signals from an IEEE 802.11 Wireless Local Area Network (WLAN) in an indoor environment, where no global-positioning system (GPS) is available.

In recent works, probabilistic approaches to activity recognition [Blaylock and Allen 2003; Charniak and Goldman 1993; Han and Veloso 2000] have been shown to be more robust and adaptable in recognizing an agent's goals and plans as compared to logic-based approaches [Kautz and Allen 1986; Lesh and Etzioni 1995]. However, most of these works have been restricted to high-level inferences. Only in recent years, have attempts been made to integrate high-level behavior inference with low-level sensor modeling. The main challenge is to bridge the gap between low-level sensor data and high-level activities through statistical models such as hidden Markov models (HMMs) [Bui et al. 2002] and dynamic Bayesian networks (DBNs) [Liao et al. 2004; Yin et al. 2004]. An overriding theme of these works has been to bridge the gap between low-level sensor data and high-level human activities through the inference of locations.

A traditional approach to activity recognition is through a location-based sensor model, where the location sequences are first estimated from low-level sensor data. High-level activities and goals are then recognized via the estimated location sequences. This approach can be referred to as Location-Based Activity Recognition (LAR). However, a major challenge of LAR is to predict the locations accurately and robustly when an indoor wireless environment changes from time to time, much calibration data must be obtained as the training data

for building the LAR model. Particularly, when there is much noise involved in the sensor data, the performance of the LAR model can degrade due to a decreased accuracy in predicted indoor locations.

To address this problem, in this article, we propose an alternative probabilistic approach for activity recognition, which is referred to as Segmentation-Based Activity Recognition (SAR). Our intuition is that although accurate prediction of individual locations of a moving agent might be useful in recognizing location-based activities, there is a large class of a person's activities in a building where each activity is more or less rough-grained in that the precise location information is not needed. Instead, a rough idea of the general trends of a user's movements is sufficient for activity recognition. This observation allows us to focus on the patterns of an activity as a whole, rather than each individual element of a pattern. For example, on seeing that a certain sensor's signal reading is gradually increasing while another, far away sensor's reading is gradually decreasing, we can recognize that the agent is moving towards the first sensor, without having to know the precise location of the agent at each moment. Using the SAR approach, high-level activities can be directly inferred from sequences of primitive signal segments. The resulting model is a high-level activity recognition model that bypasses the location level and directly bridges between sensor data and activities.

In this article, we show that using the SAR model has the following great advantages over the LAR model. First, the calibration effort for activity recognition can be significantly reduced because it only needs the user traces with goal labels as input. This corresponds to much less data collection effort as compared to the LAR approach, because the LAR approach requires that the signals collected at each location must be coupled with a human-input location label. We show that despite much less calibration effort, the SAR approach can still achieve comparable recognition accuracy when compared to the LAR approach. Second, the SAR approach is more robust than the LAR approach with respect to the observation noise involved in the user traces. Third, the LAR approach can be easily extended to recognize fine-grained indoor activities that require motion skills in one location, by incorporating additional information from motion sensors [Loke 2005].

In particular, in building the SAR model, the input of the segmentation algorithm consists of a collection of user traces, each corresponding to a sequence of signals received by the mobile device when a user is moving in a wireless environment. Each trace is a multivariate time series associated with a goal label, which indicates the objective the user is achieving in that trace. Taking the training data as input, we apply an EM algorithm to obtain a probabilistic segmentation model, so that the signal segments represented as motion patterns, along with their transition probabilities can be learned for each activity. In the online phase, given the model learned from the training data, a new user trace can be automatically partitioned into signal segments as it is received by a mobile device, and the corresponding goal can be recognized.

The rest of the article is organized as follows. Section 2 formulates the problem of activity recognition. Section 3 reviews previous work related to our problem. Section 4 discusses the location-based activity recognition algorithm.

Section 5 presents our new segmentation-based algorithm for activity recognition. Section 6 evaluates our proposed algorithm through extensive experiments on a real data set collected from a WLAN environment. Section 7 concludes the article and discusses directions for future work.

2. PROBLEM STATEMENT AND APPLICATION DOMAIN

Let Y be an observed trace that records a user's activities in a wireless environment. The trace, Y, is a multivariate time series that consists of T samples $Y_1, Y_2, \ldots, Y_T$. Each sample, Y_i, is an m-dimensional signal vector $Y_i =< Y_i^1, Y_i^2, \ldots, Y_i^m >$ received at the time instant t_i, where m is the number of sensors (APs). Each trace, Y, is associated with one goal label, $G_i \in \mathbb{G}$, where G_i denotes the goal aimed at by the user in the trace, and $\mathbb{G}$ is the set of all possible goals in a certain environment. The problem of activity recognition can be defined in two phases:

(1) In the offline phase, suppose that we are given a collection of training sequences, $\mathcal{D} = \{Y^{(1)}, Y^{(2)}, \ldots, Y^{(N)}\}$, which consists of user traces, $Y^{(i)}$, together with their associated goal label, G_i. Our objective is to train a model of how each high-level activity would lead to possible sequences of observations. Specifically, we need to learn an activity recognition model, $\mathcal{M}_A$, for specifying the conditional probability distributions, $P(Y^{(i)}|G_i)$.

(2) In the online phase, given a newly observed user trace Y with unknown goals, we would like to compute a predicted goal G^* for the trace Y, such that

$$G^* = \arg\max_{G_k} P(G_k|Y, \mathcal{M}_A).$$

In the discussion that follows, we use the terms activity recognition, plan recognition, and goal recognition interchangeably.

We now illustrate the problem of activity recognition in an indoor WLAN environment. The layout of the environment is shown in Figure 1. In the figure, four APs are marked with solid circles in this area, each of which is identified by its unique Media Access Control (MAC) address.

While a user with a mobile device performs different activities in this environment, the mobile device can periodically record signal-strength measurements from the APs. After collecting user traces and labeling each trace with its intended goal, we thus obtain a database of a user's behavior traces, as given in Table I. Two examples of these user traces are: G_4 = *Entrance3-to-Office* and G_5 = *Office-to-Print-in-Room1*. Each user trace is represented as a sequence of signal-strength values, where each element of the trace is a signal-strength vector. At each time instant t_i, the vector consists of three pairs of MAC addresses and signal-strength measurements received from the corresponding APs. For illustration, we can see from Table I that, at the time instant t_2, the signal-strength values received from the three APs in the first user trace are -81, -77, -64, and -41, respectively.

In the training phase, an activity recognition model can be learned to model the mappings between signal-strength values and activities based on these user traces. Then in the online phase, when new signal-strength values are

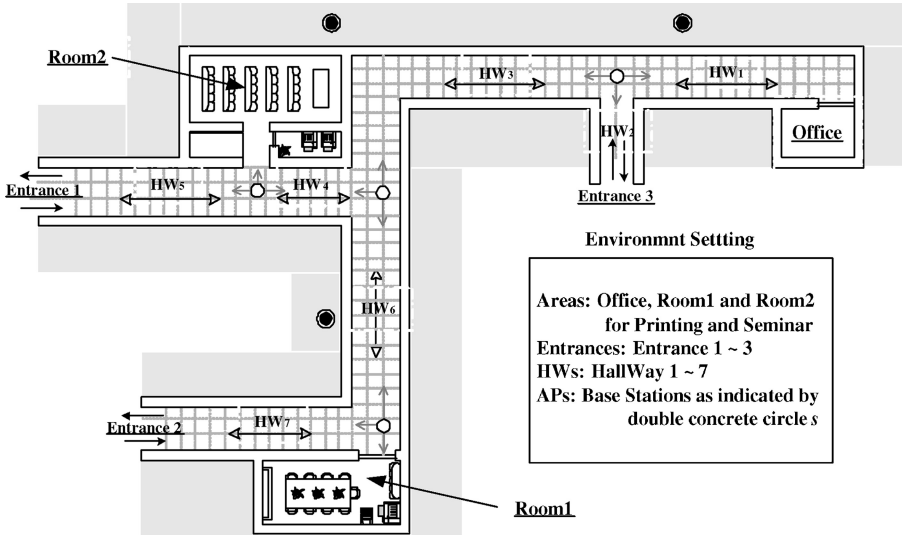

Fig. 1. The layout of the office area of the department of computer science and engineering at the Hong Kong University of Science and Technology.

Table I. Examples of User Traces

Trace #	Observed Signal Sequences				Goal #
	t_1	t_2	$\cdots$	t_n	
1	$(AP_1 : -80)$ $(AP_2 : -78)$ $(AP_3 : -62)$ $(AP_4 : -37)$	$(AP_1 : -81)$ $(AP_2 : -77)$ $(AP_3 : -64)$ $(AP_4 : -41)$	$\cdots$ $\cdots$ $\cdots$ $\cdots$	$(AP_1 : -68)$ $(AP_2 : -84)$ $(AP_3 : -81)$ $(AP_4 : -65)$	G_1
2	$(AP_1 : -66)$ $(AP_2 : -85)$ $(AP_3 : -84)$ $(AP_4 : -67)$	$(AP_1 : -63)$ $(AP_2 : -81)$ $(AP_3 : -78)$ $(AP_4 : -59)$	$\cdots$ $\cdots$ $\cdots$ $\ldots$	$(AP_1 : -78)$ $(AP_2 : -75)$ $(AP_3 : -61)$ $(AP_4 : -43)$	G_2

received from the APs while a user is moving, we wish to infer the most likely activity that he is engaging in. For example, when a new user trace $< (AP_1 : -79)(AP_2 : -75)(AP_3 : -64)(AP_4 : -40) >< (AP_1 : -80)(AP_2 : -75)(AP_3 : -65)(AP_4 : -43) > \ldots$ is observed, we would like to infer which goal the user is most likely pursuing, G_1 or G_2, in the environment.

The focus of our work is to build an activity recognition model while incurring less calibration effort for data collection. As we discussed in Section 1, many traditional approaches to activity recognition first build a location-based sensor model and then use the sensor model to construct a hierarchical activity recognition model. In order to construct the sensor model, manual effort is required to label the signal-strength samples received at each physical location. However, such approaches have a number of limitations. First, collecting the training data itself is very labor intensive, as sufficient samples must be calibrated for each location. For example, suppose that in a small office building environment with 100 selected physical locations, 100 samples are collected at each location, with one sample per second. To construct a location-based sensor

model, several hours are typically required by a person to collect and calibrate the data, even for this relatively small area. The calibration problem is more serious when the area of interest, such as a shopping mall, is very large, where spatially high-density calibration is needed. Second, even when the location labels are obtained, such data may need repeated updates due to the dynamically changing environment. For example, in an office building environment, due to the change of temperature, moisture, and lighting, signal-strength values received at the same location may change drastically from the night to the daytime. Therefore, it is very expensive to repeatedly collect training data for maintaining the accuracy of a sensor model. Consequently, location-based activity recognition has been difficult to accomplish in an indoor environment.

A naive way to reduce the calibration effort is to simply reduce the number of samples collected at each physical location. However, this simple sample-reduction approach may lead to an inaccurate sensor model, which subsequently limits the accuracy of activity recognition. To illustrate this point, we applied a hierarchical activity recognition algorithm as described in our previous work [Yin et al. 2004] (which we will review in Section 4), where we proposed a two-level architecture of Bayesian models to infer a user's goals based on a location-level sensor model. The sensor model in this approach stores a signal-strength distribution at each sampled location, which allows the locations to be inferred from the received signal-strength values at the mobile device. However, when the training samples calibrated for each location are reduced, the signal-strength distributions can easily get skewed and subsequently the performance of activity recognition degrades remarkably. Specifically, when the samples calibrated at each location are reduced from 100 to 50, the recognition accuracy decreases from 92.7% to 78.7%. Therefore, what would be ideal is allowing an activity recognition model to accurately recognize goals without tediously building a location-based sensor model. In this article, we present a user-trace segmentation based approach to achieve this objective.

3. RELATED WORK

In this section, we review previous work related to our activity recognition problem in three parts. In Section 3.1, we review research in plan recognition that includes both logic-based plan recognition and probabilistic plan recognition. In Section 3.2, we review different approaches to constructing sensor models for estimating locations from signals. In Section 3.3, we discuss other related probabilistic models used for activity recognition. Finally, in Section 3.4, we review related work in time series analysis and segmentation in the data mining area.

3.1 Plan Recognition

In the area of artificial intelligence, recognizing a user's high-level activities has traditionally been a major focus of plan recognition [Blaylock and Allen 2003; Charniak and Goldman 1993; Goldman et al. 1999]. Plan recognition is the process of inferring an agent's goals and plans from observations. Typically, the observations can be actions performed by an agent, or observable effects

of the actions. In previous literature, there are two categories of methods for solving the problem of plan recognition: logical approaches and probabilistic approaches.

Given a set of observations, logical approaches search a space of possible plan hypotheses for candidate plans and goals that account for the observations. To form the search space in a given domain, a well-defined plan library about the domain knowledge is required. Kautz's event hierarchy formalism is one of the most famous frameworks for performing plan recognition [Kautz and Allen 1986; Lesh and Etzioni 1995]. In this framework, actions and plans are uniformly referred to as events. An event hierarchy is a collection of first-order statements that represent the abstraction and decomposition relations among various types of events. After each observation is received, deductive inference is used to reason from a set of observations to form hypotheses about possible top-level plans and future actions. Recently, logic programming has been widely used to represent and reason about the users' situations for building context-aware systems [Loke 2005]. While logical approaches provide a formally elegant solution for plan recognition, the most serious problem is that they are not able to represent uncertainty. Therefore, there is a lack of methods used for choosing one particular plan over another, as long as the observations can be explained by both of them.

Probabilistic approaches, on the other hand, consider plan recognition as a process of inference under uncertainty. Under the probabilistic framework, an agent can compute a probability distribution over all the candidate plans. This therefore allows the recognizer to distinguish among equally possible, but unequally plausible explanations for the observations. Over the years, various probabilistic models have been proposed to reason about an agent's plans and goals [Blaylock and Allen 2003; Charniak and Goldman 1993; Han and Veloso 2000; Pynadath and Wellman 2000]. A Bayesian network was used for plan recognition in story understanding [Charniak and Goldman 1993]. In the UNIX domain, a corpus-based N-gram model was proposed to predict a user's goal from a given sequence of command actions [Blaylock and Allen 2003]. In addition, other advanced stochastic models for activity recognition were proposed, such as dynamic Bayesian networks (DBNs) [Albrecht et al. 1998] and probabilistic state dependent grammars [Pynadath and Wellman 2000]. However, most of these works were restricted to high-level inference in which action sequences were taken as input, whereas the challenge of dealing with low-level sensor data has not been addressed.

In recent years, there has been an increasing interest in recognizing high-level human behavior from various types of sensors [Bui et al. 2002; Liao et al. 2004; Muhlenbrock et al. 2004; Yin et al. 2004]. This gives birth to a newly emerging research problem—sensor-based activity recognition—which involves pervasive computing and artificial intelligence. In an outdoor environment, a DBN was applied to estimate a person's locations and transportation modes from logs of GPS measurements [Liao et al. 2004; Patterson et al. 2003]. Since the GPS technology can directly provide physical locations and even velocity information, location-based activity recognition can be enabled in which calibrated data with locations are not required. In an indoor environment, Yin

et al. [2004] proposed a two-level architecture of Bayesian models to infer a user's goals based on RF signal strength received from a wireless network. Muhlenbrock et al. [2004] applied the maximum a posteriori (MAP) estimation to infer the users' activities and availabilities from a variety of sensor data. These works explicitly relied on training a location-based sensor model to estimate locations from signals, through which goals can be recognized at the high level. Therefore, the calibrated data labeled with locations are part of the input for training. Likewise, in the vision community, abstract hidden Markov models (AHMMs) [Bui et al. 2002; Nguyen et al. 2003] were introduced to infer a person's goals from camera data. Similarly, a location-based sensor model was required to be trained so as to convert camera images into locations.

In summary, an overriding theme of these works has been to bridge the gap between high-level activities and low-level sensor readings through inference about locations. As a result, a location-based sensor model at the low level is needed to be trained offline in order to enable higher-level activity recognition to be performed online. This requires a lot of manual calibration, which is quite expensive and time-consuming. Therefore, in this work, we focus on the problem of how to reduce the calibration effort in building an activity-recognition model.

3.2 Location Estimation

In the pervasive computing area, significant research work has been done on determining and tracking a user's locations from sensor data to provide location-based services. Examples include the use of GPS [Enge and Misra 1999], wide-area cellular-based systems [Tekinay 1998], ultrasonic-based systems [Priyantha et al. 2000], infrared-based systems [Want et al. 1992], and RF-based systems [Bahl and Padmanabhan 2000; Bahl et al. 2000; Fox et al. 2002; Youssef et al. 2003; Ladd et al. 2002]. Among these systems, the class of RF-based systems, which utilize an underlying wireless network, such as 802.11b, to estimate the locations of users, has gained more attention recently, especially for indoor applications. Unlike infrared-based systems, RF-based techniques can provide more ubiquitous coverage, and do not require additional expensive hardware for location determination, thereby enhancing the value of the wireless network.

In general, RF-based systems function in two phases: an offline training phase and an online localization phase. In the offline phase, a sensor model is built by tabulating the signal-strength values received from the APs at selected locations in the area of interest. These values comprise a sensor model of the physical region, which is compiled into a nonparametric or parametric prediction model for the online phase. In the online phase, the real-time signal-strength samples received from the APs are used to search the sensor model to estimate the current location based on the learned model.

RF-based location-estimation techniques can be classified into two broad categories: nonparametric techniques and parametric techniques. Nonparametric techniques [Bahl and Padmanabhan 2000; Bahl et al. 2000; Smailagic and Kogan 2002] apply deterministic inference methods to estimate a client's

location. For example, the RADAR system by Microsoft Research [Bahl and Padmanabhan 2000; Bahl et al. 2000] uses the k-nearest neighbor method to infer a user's location. Each signal-strength sample is compared against the sensor model in the online phase. The coordinates of the k best matches are averaged to give the location estimation.

Parametric techniques, on the other hand, construct the signal-strength distributions over different locations in the sensor model and use probabilistic inference methods for localization [Ladd et al. 2002; Roos et al. 2002; Youssef and Agrawala 2004; Youssef et al. 2003]. The robotics-based location sensing system [Ladd et al. 2002] applies Bayesian inference to compute conditional probabilities over locations based on received signal-strength samples from various APs. The spatial constraints of a user's movements are also used in a post-processing step to refine the estimation. In Youssef et al. [2003], locations in the area are preclustered into groups so as to reduce the computational cost of searching the sensor model. Chai and Yang [2005] make use of user traces that are not labeled with locations, in an effort to reduce the calibration effort for building the sensor model. The core of all these techniques is the use of Bayesian inference to compute posterior probabilities over locations conditioned on received signal-strength values.

However, despite the large amount of work in this area, there has been little study on modeling and inferring a user's activities from sequences of sensor data. Yet as observed in Patterson et al. [2003], having a good understanding about a user's high-level behavior patterns and goals will help in estimating the user's current locations.

3.3 Other Relevant Probabilistic Models

Our probabilistic segmentation model is closely related to two existing probabilistic models: hierarchical hidden Markov models (HHMMs) [Bui et al. 2004; Fine et al. 1998] and switching linear dynamic systems (SLDSs) [Bregler 1997; Ghahramani and Hinton 1998; Murphy 1998; Oh et al. 2005; Pavlović and Rehg 2000; Pavlović et al. 1999, 2000].

The hierarchical hidden Markov model (HHMM) is an extension of the standard hidden Markov model (HMM) to include a hierarchy of the hidden states. Therefore, an HHMM generates observation sequences by a recursive activation of one of the substates of a state in the model. Motivated by the inside-outside algorithm [Lari and Young 1990] for probabilistic context-free grammars (PCFGs), Fine et al. [1998] presented a method for inference and expectation-maximization (EM) parameter learning in the HHMM with the complexity of $O(NT^3)$, where T is the length of the observation sequence, and N is the total number of hidden states. Nevertheless, the state hierarchy in the original HHMM is restricted to a tree structure, which prohibits two different states from having the same child. To address this problem, Bui et al. [2004] proposed a general HHMM in which the state hierarchy can be a lattice, which allows arbitrary sharing of substructures.

The switching linear dynamic system (SLDS) can be viewed as a generalization of an HMM in which each switching state is associated with a linear

dynamic system (LDS). An SLDS model can represent the nonlinear dynamic behavior of a complex system by switching among a set of LDS models over time. Therefore, an SLDS is more appropriate to represent and model complex human behavior than an HHMM. However, offsetting this advantage is the fact that exact inference in an SLDS is intractable, which in turn complicates parameter learning via the EM algorithm [Murphy 1998; Pavlović and Rehg 2000; Pavlović et al. 2000]. Therefore, there have been many efforts to derive approximate inference techniques for SLDS models. Previous work on approximate inference in SLDS models includes approximate Viterbi inference [Pavlović et al. 1999], generalized pseudo Bayesian (GPB2) inference [Pavlović et al. 2000], structured variational methods [Ghahramani and Hinton 1998; Pavlović et al. 2000], and Markov-Chain Monte-Carlo (MCMC) methods [Oh et al. 2005].

3.4 Time Series Analysis and Segmentation

Our work is also related to time series analysis and segmentation in the data mining area. On the surface, segments can be obtained by applying existing time-series analysis algorithms. However, on closer examination, this is not the case. In the data mining area, many previous works [Chiu et al. 2003; Oates 2002; Zaki 2001] focused on finding frequent patterns based on the idea of finding frequently recurring segments in a time series. In our problem, however, the target segments may not correspond to frequent patterns; thus frequency is not a target metric. Our problem is also different from general time-series segmentation, which aims at partitioning data sequences into nonoverlapping, internally homogeneous segments. For example, Himberg et al. [2001] applied a dynamic programming algorithm to identify context information by segmenting the time series received from sensors. The Sliding Window algorithm [Keogh et al. 2001] was also used to partition the time series given a user-specified threshold. It is attractive for solving our problem because of its simplicity. However, an appropriate threshold for the window size needs to be specified, which is hard to determine. Most of these works followed an unsupervised framework, which relied on weak measures of quality based on information theory. In contrast, our objective is more specific; it is to segment data sequences such that goals can be accurately recognized. Thus, the segments that we discover are highly dependent on goals and activities, which can be leveraged in a supervised way to recognize goals and activities.

Several works in behavior recognition are also related to our problem. Czielniak et al. [2003] learned motion patterns from a collection of trajectories using clustering, but the trajectory-segments need to be constructed by hand. Li et al. [2002] applied a linear dynamic model to learn motion textons from a single human-dance sequence, which can then be used to generate new animation sequences. However, this work followed an unsupervised framework that cannot be directly adapted to recognize high-level goals given a collection of training sequences. Moreover, Peursum et al. [2004] employed a hidden Markov model to segment individual actions from a stream of human motion, but it requires human-supplied action labels as part of the input during the learning

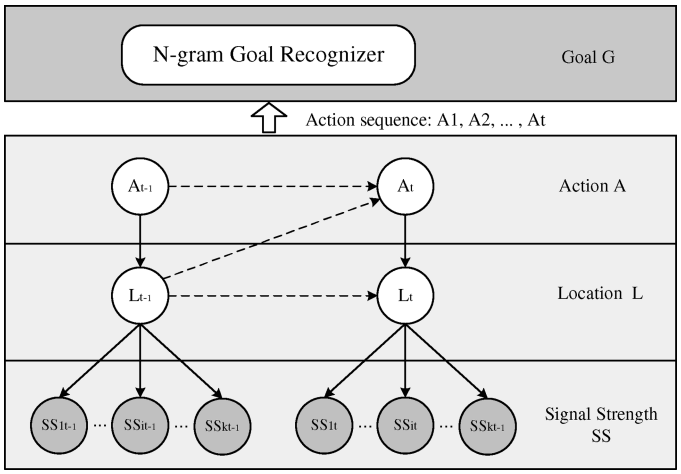

Fig. 2. Location-based activity recognition model (LAR).

process. In contrast, we aim at automatically learning semantically meaningful patterns from training sequences for activity recognition.

4. LOCATION-BASED ACTIVITY RECOGNITION

In this section, we present a location-based activity-recognition model (LAR). This model is adapted from the two-layer recognition model [Yin et al. 2004], which relies on a sensor model for location estimation at the lowest level. The architecture of the LAR model is shown in Figure 2. This figure shows two time slices that are numbered t and $t-1$, respectively. In the figure, the shaded nodes, SS, represent the strength variables of signals received from the APs, which can be directly observed. All the other variables—the physical location, L, of the user, the action, A, the user is carrying out and the goal, G, the user is pursuing—are hidden, with the values to be inferred from the raw signals. The LAR model is a two-level Bayesian architecture in which a user's goals, G, can be inferred from signal-strength measurements, SS. At the lower level, based on a sensor model, a DBN is applied to estimate a user's actions, A, from signal-strength measurements, SS. On top of this, an N-gram recognizer is adopted to predict goals, G, from action sequences at the high level.

An important component of the LAR model is the location-based sensor model for estimating locations, L, from signal-strength measurements, SS. To build this model, we model the world as a finite location space $\mathbb{L} = \{l_1, \ldots, l_n\}$, with a set of observations. Each observation, Y_j, along an observation sequence, $Y = \{Y_1, \ldots, Y_j, \ldots\}$, is represented as an m-dimensional signal-strength vector $Y_j =< Y_j^1, Y_j^2, \ldots, Y_j^m >$, where m is the total number of APs, and Y_j^k, $1 \leq k \leq m$, represents the strength of the signal received from the k^{th} AP. Since signals are noisy and a single scan will probably miss some APs, we take an average over the signal-strength measurements every second and take that as an observation. Formally, we define the location-based sensor model as a

predictive model, which uses the conditional probabilities $P(Y_j|l_i)$: the likelihood of observing the signal-strength vector Y_j at location l_i.

To construct the sensor model, signal-strength measurements are collected at each location, l_i, in the offline phase. After the data are calibrated, we build a histogram of observations for each AP at each location l_i. This is done by constructing a conditional probability distribution $P(Y_j^k|l_i)$, which is the probability of receiving the signal-strength value Y_j^k from the k^{th} AP at location l_i. By further making an independence assumption among signals from different APs, we multiply all of these probabilities to obtain the conditional probability of receiving a particular signal-strength vector, Y_j, at location l_i, as follows:

$$P(Y_j|l_i) = \prod_{k=1}^{m} P\left(Y_j^k|l_i\right). \tag{1}$$

Based on the sensor model, we can learn a DBN model from a collection of training traces, $\mathcal{D}$. The model parameters are estimated using an expectation maximization (EM) algorithm, in which the inference at the E-step is achieved by the junction tree algorithm [Murphy 2002]. We use the constructed sensor model to initialize the conditional probability distribution $P(Y_j|l_i)$ in the DBN model. The distribution is updated via the EM algorithm while learning other parameters of the DBN model.

After the DBN model is learned, given a sequence of signals $Y_1, Y_2, \ldots, Y_t$ obtained up to time t, we can use the inference algorithm to compute the most probable action sequence $A_1, A_2, \ldots, A_t$. At the next level up, we can further infer the most likely goal from actions. In particular, given an inferred sequence of actions obtained so far, $A_1, A_2, \ldots, A_t$, find the corresponding most likely goal, G^*:

$$G^* = \arg\max_{G_k} P(G_k|A_1, A_2, \ldots, A_t) = \arg\max_{G_k} P(G_k|A_{1:t}). \tag{2}$$

By applying Bayes' Rule, this formula becomes:

$$G^* = \arg\max_{G_k} \frac{P(A_{1:t}|G_k)P(G_k)}{P(A_{1:t})} = \arg\max_{G_k} P(A_{1:t}|G_k)P(G_k). \tag{3}$$

In this equation, the likelihood $P(A_{1:t}|G_k)$, can be calculated by adopting an N-gram model. The complexity of inference for the LAR algorithm is $O(K^w T)$, where K is the maximum number of possible values that each discrete hidden state can take, w is the maximum size of cliques constructed using the junction tree algorithm, and T is the length of an observation sequence.

The advantage of this two-layer recognition model is that it is more flexible and adaptive to changes, than a monolithic DBN model. In this model, any changes to the sensor model or the behavior model will not cause the whole model to be reconstructed. For example, when different types of sensors are employed in, or added to, the environment, the sensor model has to be reconstructed, while a user's behavior, such as plans and goals, remain unchanged; in contrast, when a user's plans and goals keep changing in the environment, the behavior model has to be retrained over time while the sensor model remains the same. In addition, the recognition accuracy of the two-layer model is

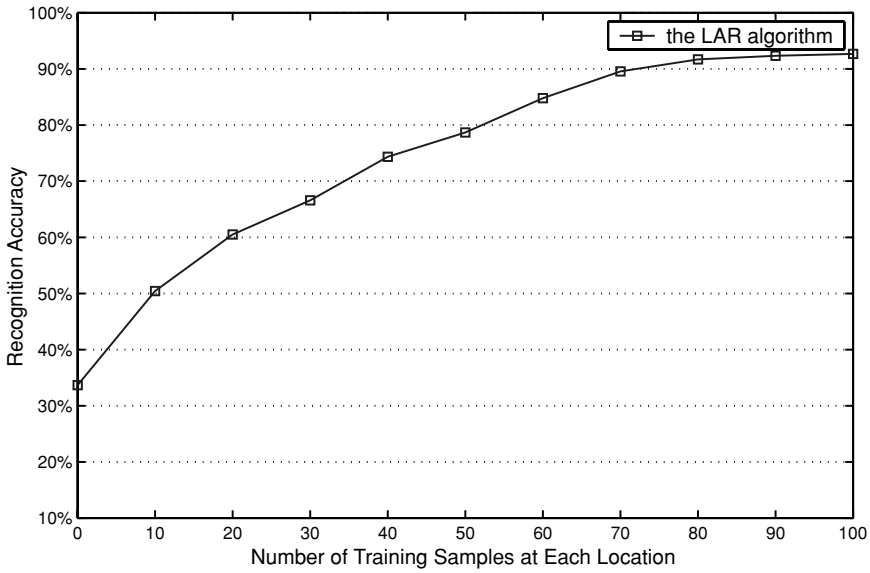

Fig. 3. Recognition accuracy vs. number of training samples at each location.

comparable to that of a monolithic DBN model. Interested readers are referred to Yin et al. [2004] for details.

In our evaluation of the LAR model, we model the environment as shown in Figure 1 as a space of 99 locations, each representing a $1.5m \times 1.5m$ grid cell. To train the location-based sensor model, we collected 100 signal-strength samples at each location, one per second. That is, we needed to spend 100 seconds at each of the 99 locations and label each sample by its corresponding location. The whole process of calibration took us about three hours to finish. In the following, we analyze the dependency of the LAR model on the location-based sensor model.

To see the impact of calibration on recognition accuracy, we recorded the accuracy of the LAR model for different numbers of training samples calibrated at each location. These samples are used as the training data to train the sensor model. We ran the experiments through a three-fold cross validation over 864 user traces collected for 19 different goals in the environment (as shown in Figure 1). For one of the 19 goals, we divided the user traces into three folds; two folds were used for training and one for testing. The recognition accuracy for each goal is therefore averaged over three runs. In this experiment, we varied the number of training samples labeled for each location. Initially, the number of training samples calibrated for each location is set to be zero. At this point, the only statistical knowledge we have is from the traces. After that, we increase the number by ten samples per location to simulate the effort of gradually increasing the calibration effort. Figure 3 shows the recognition results with respect to different numbers of training samples ranging from 0 to 100. Each value plotted in the figure corresponds to the overall recognition accuracy averaged over 19 goals.

From Figure 3, an interesting observation is that when the training samples are scarce, the amount of calibration data has a significant influence on accuracy. In particular, when there are no training data for building the sensor model, that is, when only user traces are used for training, the recognition accuracy of the LAR model is only 33.7%. In this case, we initialize both the sensor model and the transition matrix to be random, where only the trace information contributes to distinguishing goals in the training phase. Therefore, the EM algorithm can be easily trapped in a local minimum due to the random initialization. Furthermore, when the number of training samples calibrated at each location increases to ten, the recognition accuracy increases by 49.6% to 50.4%. This indicates that, having more training data to build the sensor model can significantly improve the performance of the LAR model. When the number of training samples continues to increase, the recognition accuracy still increases, but the improvement is less significant when more training samples are available. For example, when the number of training samples increases from ten to twenty, the recognition accuracy increases by 20.0% to 60.5%. Finally, the recognition accuracy increases to 92.5% when 90 training samples at each location are used to build the sensor model. This corresponds to the calibration effort of about three person-hours. However, the recognition accuracy starts to saturate when even more training samples are used. This is mainly because signal propagation in an indoor environment suffers from severe multipath fading effects, which leads the received signal strength to be uncertain and noisy. As a consequence, location estimation cannot achieve perfect accuracy, and thus the performance of the LAR model is limited.

In summary, the performance of the LAR model depends critically on the number of calibrated samples for each location used to build the sensor model. Reducing the calibrated samples can severely degrade the performance of the LAR model due to an inaccurate sensor model at the lowest level. Consequently, the LAR model has limited ability to reduce the calibration effort for activity recognition. The situation is more serious when the area of interest is vary large, where the calibration of spatially high density is required. Therefore, it is desirable to design an activity-recognition algorithm that requires less calibration effort, while the recognition accuracy can still be guaranteed.

5. SEGMENTATION-BASED ACTIVITY RECOGNITION: OUR NEW APPROACH

Instead of first building a location-based sensor model, our segmentation-based approach recognizes goals directly from sequences of signal segments. In this section, we present this new approach in detail. A preliminary version of this work has been presented in Yin et al. [2005a].

5.1 Overview of Our New Approach

We first illustrate the intuition behind our new approach using Figure 4. For simplicity, Figure 4(a) gives an example of three user traces received from a particular AP in our wireless environment, where the x-axis corresponds to time and the y-axis corresponds to the value of signals received at each time.

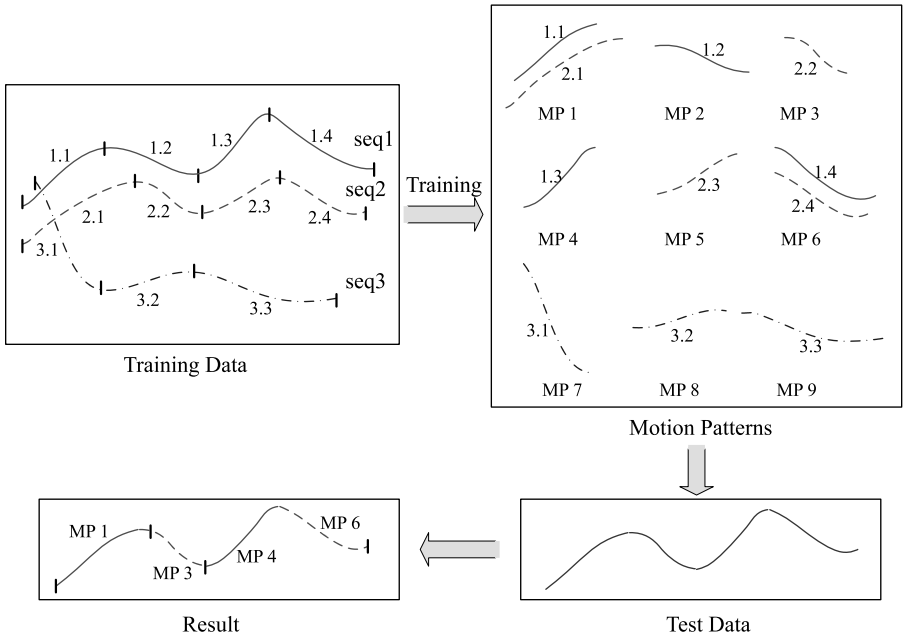

Fig. 4. Flow chart of our new approach.

Sequences *seq1* and *seq2* denote the user traces for achieving a goal G_1, and the sequence *seq3* denotes the user trace to achieve a goal G_2. By performing segmentation, each user trace can be partitioned into several signal segments, where each segment consists of signals that exhibit consistent characteristics in the signal space, in terms of the magnitude and trend of signals. For example, the trace *seq1* can be partitioned into four signal segments, namely, signal segments that are labeled as 1.1, 1.2, 1.3, and 1.4. Each of these segments consists of signals that demonstrate consistent behavior, such as increasing in value, or decreasing in value. Notice that this example is intentionally simplified by drawing the user traces as a one-dimensional vector sequence. However, in reality, there can be a large number of APs under our consideration. In this case, the segmentation cannot be easily displayed on a two-dimensional picture for visualization, and simple shape-based segmentation algorithms may not apply.

Figure 5 visualizes a user trace collected in our wireless environment shown in Figure 1. For illustration, we show the trace only using the signals from four APs in the figure. This trace, corresponding to the goal *Entrance2-to-Office*, records the movements of a user from *Entrance2* to his office. As we can see from the figure, this trace is partitioned into five segments and each segment represents a typical motion pattern. For example, the segments *seg3* and *seg4* indicate that the user walks through HW3, because the signals from AP1 significantly increase and then decrease, while the signals from the other APs gradually decrease; *seg5* indicates that the user stays in his office because the signals are relatively stable.

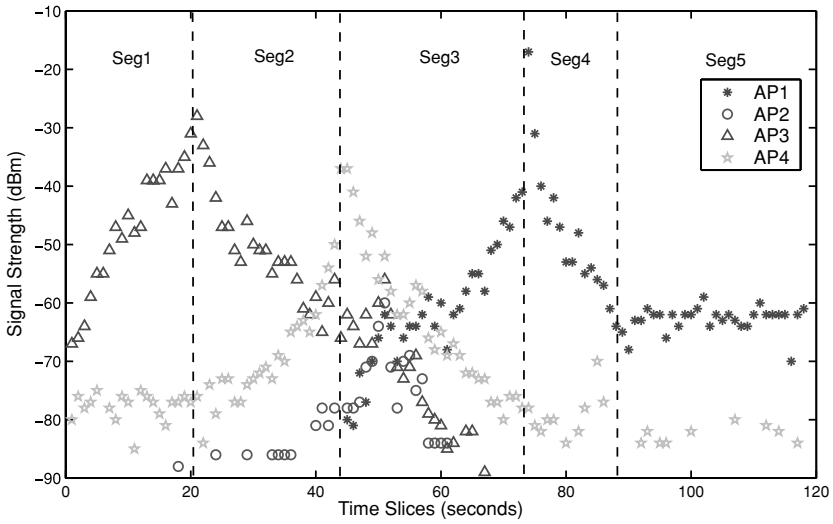

Fig. 5. A user's behavior trace collected from four APs.

Our next task is to associate the signal segments with the user activities. Our basic assumption is that the segments can serve as atomic units of a user's movements, and that their collective composition determines the user goals. Intuitively, for a set of user traces that result from the same activity, the signal segments from different sequences may share the same characteristics with each other. To capture this notion, we define a motion pattern to represent similar signal segments. For instance, the eight segments generated from sequences *seq1* and *seq2* can be represented by six motion patterns from *MP1* to *MP6,* as shown in Figure 4(b), where *MP1* represents segments 1.1 and 2.1.

For each activity, the motion patterns can be learned from training traces using the algorithm we describe in Section 5.3. For example, the goal G_1 is associated with motion patterns from *MP1* to *MP6*, and the goal G_2 is associated with motion patterns from *MP7* to *MP8.* After learning the motion patterns for each activity in the training phase, we can use these patterns to classify new sequences in an online phase. As shown in Figures 4(c) and 4(d), since the test sequence can be well represented by the motion patterns associated with the goal G_1: *MP1*, *MP3*, *MP4,* and *MP6*; we can therefore infer that the user is likely to be achieving the goal G_1.

In the following, we explain our new approach in three key aspects. Section 5.2 discusses how to represent user traces using a probabilistic segmentation model. Section 5.3 presents how to learn the model parameters for each activity in the offline training phase. Section 5.4 discusses how to perform goal recognition in the online phase.

5.2 Probabilistic Segmentation Model

Given an observed signal sequence Y, which consists of T samples, Y_1, $Y_2, \cdots, Y_T$, and a corresponding goal label G, we propose a probabilistic

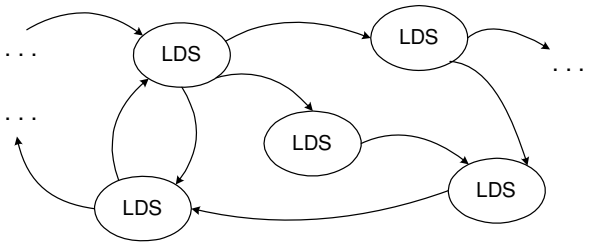

Fig. 6. Probabilistic segmentation model.

segmentation model to represent the observation sequence Y based on a user's activity. In our model, suppose that there are N_m motion patterns $\mathcal{P} = \{P_1, P_2, \ldots, P_{N_m}\}$, which are generated by the underlying hidden activities. Each motion pattern P_i, $1 \leq i \leq N_m$, consists of signals that exhibit similar characteristics, in terms of the magnitude and trends in the signal space. The values of signals in each motion pattern demonstrate consistent behavior, such as increasing or decreasing at a similar rate. Our objective is to partition the observation sequence Y, into consecutive signal segments, such that each segment can be represented by one of the N_m motion patterns. Note that multiple segments can be represented by the same motion pattern. The relationship between any pair of motion patterns can be described by the probability of switching from one motion pattern to another. As illustrated in Figure 6, we first use a motion pattern, represented by a linear dynamic system (LDS) model, to capture the local linear dynamics involved in each segment, and then adopt a transition matrix among motion patterns to model the global nonlinear dynamics contained in the stochastic process.

We adopt a switching linear dynamic system (SLDS) [Murphy 1998; Pavlović et al. 2000] to model the dynamics of such a nonlinear stochastic process. The system can be described using the following state-space equations:

$$\begin{cases} X_t = AX_{t-1} + w_t, & w_t \sim \mathcal{N}(0, Q), \\ Y_t = CX_t + v_t, & v_t \sim \mathcal{N}(0, R), \end{cases} \tag{4}$$

for an LDS model to represent the local linear dynamics, and

$$M_{ij} = P(S_t = i | S_{t-1} = j), \tag{5}$$

$$\pi_i = P(S_1 = i), \tag{6}$$

for the switching model to capture the global nonlinear dynamics.

In these state-space models, $X_t \in \mathbb{R}^k$ denotes the hidden state of the LDS model, and $Y_t \in \mathbb{R}^m$ denotes the observed signal measurements at time t. We assume that the sequence of X's satisfies a first-order Markov property, which states that the future states depend only on the current state, and at each time step, the observation Y_t is generated from the hidden state X_t. An LDS model comprises a linear Gaussian state model and a linear Gaussian observation

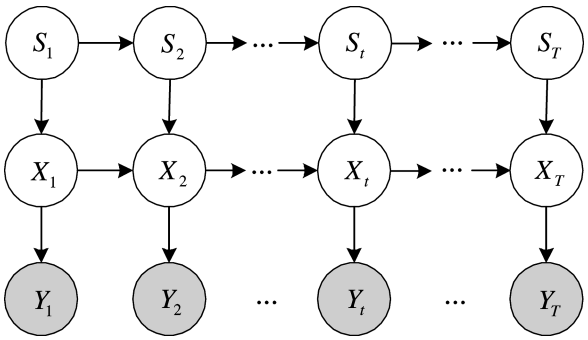

Fig. 7. The graphical model representation of the SLDS model.

model. For the state model, parameter A denotes the state transition matrix determining the mean of X_t given X_{t-1}, and w_t is the state noise represented by a zero-mean Gaussian distribution with covariance matrix Q. Similarly, for the observation model, parameter C denotes the observation matrix determining the mean of Y_t given X_t, and v_t is the observation noise represented by a zero-mean Gaussian distribution with covariance matrix R. Moreover, the state-space models are assumed to satisfy the stationary property, which indicates the model parameters are time-invariant. Thus, an LDS model can be represented by its model parameters, $\theta = \{A, C, Q, R\}$.

The switching model specifies the transition probabilities among the LDS models. The state variable S_t is a multinomial variable that can take on N_m values: $S_t \in \{1, \ldots, N_m\}$. The switching model is assumed to satisfy a discrete first-order Markov process, which is defined using the transition matrix M and an initial state distribution vector π. Since the switching state variable S_t determines which of the N_m LDS models is used at time t, the associated parameters A and Q of an LDS model depend on the value of the switching state S_t. Namely,

$$A(S_i) = A_i, \tag{7}$$

$$Q(S_i) = Q_i. \tag{8}$$

Based on the switching model, the global nonlinear dynamics involved in the user traces can be modeled accordingly.

Therefore, the model parameter of an SLDS model can be represented by $\Theta = \{A_i, Q_i, C, R, M, \pi\}$, where $1 \leq i \leq N_m$ corresponds to different LDS models. The number of LDS models, N_m, should be set properly so that the global nonlinear dynamics in user traces can be captured; later, we will explore the effect of varying the number of LDS models N_m on the performance of the SAR algorithm.

The state-space model of an SLDS model can also be represented by the graphical model shown in Figure 7. The shaded variables, Y_t, $1 \leq t \leq T$, represent the observed signal measurements at time t. The middle chain, X_t, $1 \leq t \leq T$, represents the hidden state variables of the LDS models. The discrete Markov chain, S_t, $1 \leq t \leq T$, determines which of the N_m LDS models is used

at time t. Based on this graphical structure, the joint probability P over all the variables can be factorized as:

$$P(S, X, Y) = P(S_1)\prod_{t=2}^{T} P(S_t|S_{t-1})P(X_1|S_1)\prod_{t=2}^{T} P(X_t|X_{t-1}, S_t)\prod_{t=1}^{T} P(Y_t|X_t). \tag{9}$$

5.3 Offline Model Learning Phase

Given an observation sequence Y, the model parameters Θ of an SLDS model can be learned using a maximum likelihood (ML) method, which aims to maximize the likelihood of observations, given the model parameter:

$$\hat{\Theta} = \arg\max_{\Theta} P(Y|\Theta) = \arg\max_{\Theta} \sum_{S,X} P(S, X, Y|\Theta). \tag{10}$$

Since the state variables S and X are hidden, we can use an EM algorithm [Dempster et al. 1977; Minka 1998] to solve the maximum likelihood problem. Given the observation data Y, the algorithm iterates through two steps until it converges to a local optimum:

—**E-step**: An inference algorithm is used to obtain the posterior distribution over the hidden variables S and X using a current estimate for the model parameters Θ^i:

$$f^i(S, X) \triangleq P(S, X|Y, \Theta^i). \tag{11}$$

—**M-step**: The model parameters Θ are updated by maximizing the expected log-likelihood:

$$\Theta^{i+1} \longleftarrow \arg\max_{\Theta} \langle \log P(S, X, Y|\Theta)\rangle_{f^i(S,X)}, \tag{12}$$

where $\langle\cdot\rangle_E$ denotes the expectation of a function $(\cdot)$ under a distribution E.

In the EM algorithm, inference in E-step is an important subroutine used to estimate the posterior probability of the hidden variables given the observation sequence based on the current model parameter. However, since the exact inference in E-step is intractable for SLDS models, we employ an approximate Viterbi inference algorithm [Pavlović and Rehg 2000; Pavlović et al. 2000], which computes an approximation to the sequence of S and X with the highest probability.

In the following, we explain the two major steps of the EM algorithm in detail.

5.3.1 *E-step: Approximate Viterbi Inference Algorithm.* The objective of the approximate Viterbi inference is to find the best sequence of switching states S_t, and LDS states, X_t, given the observation sequence $Y_{1:T}$, which can maximize the posterior probability in Equation (11). In order to calculate the best sequence of switching states, we need to define the quantity:

$$J_{t,i} = \max_{\{S_{1:t-1}, X_{1:t}\}} \log P(S_{1:t-1}, S_t = i, X_{1:t}, Y_{1:t}), \tag{13}$$

which represents the maximum log-likelihood of the observation sequence $Y_{1:t}$ when the switching state sequence is in state i at time t. Based on this definition, the likelihood $J_{t,i}$ can be calculated via a recursive procedure as follows:

$$J_{t,i} \;=\; \max_j \left\{ J^{t-1}_{t,i,j} + J_{t-1,j} \right\}, \tag{14}$$

where $J^{t-1}_{t,i,j}$ represents the likelihood associated with the transition from state j at time $t-1$ to state i at time t. $J_{t-1,j}$ represents the maximum likelihood of the observation sequence $Y_{1:t-1}$ when the switching state is j at time $t-1$. To actually retrieve the state sequence, we also need to keep track of the argument j, which can maximize the overall likelihood $J_{t,i}$ in Equation (14), for each switching state i at time t. Thus, we define the state transition record:

$$\psi_{t-1,i} \;=\; \arg\max_j \left\{ J^{t-1}_{t,i,j} + J_{t-1,j} \right\}. \tag{15}$$

At the end of the forward recursions, we can compute the maximum likelihood, $J_{T,i}$, and the most likely switching state sequence, $S^*_{1:T}$, based on the observation sequence, $Y_{1:T}$. Below, we discuss how to calculate the likelihood, $J_{t,i}$, recursively. For this purpose, we first define the following statistics for inference. Here the operator $\langle\cdot\rangle$ denotes expectation with respect to the posterior distribution, $P(S, X|Y)$.

$$\begin{aligned}
\hat{X}^t_{t,i} &\triangleq \langle X_t | Y_{1:t}, S_t = i \rangle, \\
\Sigma^t_{t,i} &\triangleq \left\langle (X_t - \hat{X}^t_{t,i})(X_t - \hat{X}^t_{t,i})' | Y_{1:t}, S_t = i \right\rangle, \\
\hat{X}^{t-1}_{t,i,j} &\triangleq \langle X_t | Y_{1:t-1}, S_t = i, S_{t-1} = j \rangle, \\
\Sigma^{t-1}_{t,i,j} &\triangleq \left\langle (X_t - \hat{X}^{t-1}_{t,i,j})(X_t - \hat{X}^{t-1}_{t,i,j})' | Y_{1:t-1}, S_t = i, S_{t-1} = j \right\rangle, \\
\hat{X}^t_{t,i,j} &\triangleq \langle X_t | Y_{1:t}, S_t = i, S_{t-1} = j \rangle, \\
\Sigma^t_{t,i,j} &\triangleq \left\langle (X_t - \hat{X}^t_{t,i,j})(X_t - \hat{X}^t_{t,i,j})' | Y_{1:t}, S_t = i, S_{t-1} = j \right\rangle.
\end{aligned}$$

$\hat{X}^t_{t,i}$ and $\Sigma^t_{t,i}$ denote the best LDS state estimate and state covariance estimate at time t given the observation sequence $Y_{1:t}$, when the switch state is i at time t. $\hat{X}^{t-1}_{t,i,j}$ and $\Sigma^{t-1}_{t,i,j}$ denote the one-step predicted LDS state estimate and state covariance estimate, given the observation sequence $Y_{1:t-1}$, when the switch state is i and j at times t and $t-1$, respectively. $\hat{X}^t_{t,i,j}$ and $\Sigma^t_{t,i,j}$ denote the LDS state estimate and state covariance estimate given the observation sequence $Y_{1:t}$, when the switch states are i and j at times t and $t-1$, respectively.

Considering a switch state transition from j to i, we can estimate the state X_t, and covariance Σ_t, at time t, given the observation sequence $Y_{1:t-1}$, up to time $t-1$. From Kalman filter theory [Anderson and Moore 1979], we have the following time update equations:

$$\hat{X}^{t-1}_{t,i,j} \;=\; A_i \hat{X}^{t-1}_{t-1,j}, \tag{16}$$

$$\Sigma^{t-1}_{t,i,j} \;=\; A_i \Sigma^{t-1}_{t-1,j} A_i^{'} + Q_i. \tag{17}$$

The two equations describe the forward propagation of the state estimate and covariance estimate before having accounted for the observation, Y_t, at time t. For a given switch state transition from j to i, we can also compute its associated likelihood, $J_{t,i,j}^{t-1}$, using Equation (18) [Pavlović et al. 2000]:

$$J_{t,i,j}^{t-1} = \frac{1}{2}\left(Y_t - C\hat{X}_{t,i,j}^{t-1}\right)'\left(C\Sigma_{t,i,j}^{t-1}C' + R\right)^{-1}\left(Y_t - C\hat{X}_{t,i,j}^{t-1}\right) + \frac{1}{2}\log|C\Sigma_{t,i,j}^{t-1}C' + R| - \log M_{ji}. \quad (18)$$

This calculation is based on two components: the first two terms indicate the state transition for an LDS model, and the last term indicates the switching state transition from state j to state i.

Given a new observation Y_t, at time t, the estimates about the state X_t and the state covariance Σ_t, can be filtered using the following measurement update equations:

$$\hat{X}_{t,i,j}^{t} = \hat{X}_{t,i,j}^{t-1} + K_t\left(Y_t - C\hat{X}_{t,i,j}^{t-1}\right), \quad (19)$$

$$\Sigma_{t,i,j}^{t} = \Sigma_{t,i,j}^{t-1} - K_t C\Sigma_{t,i,j}^{t-1}, \quad (20)$$

$$K_t = \Sigma_{t,i,j}^{t-1}C'\left(C\Sigma_{t,i,j}^{t-1}C' + R\right)^{-1}. \quad (21)$$

Here K_t is referred to as the Kalman gain matrix [Ghahramani 1998].

Now the likelihood $J_{t,i}$ can be calculated based on the likelihood $J_{t-1,i}$ at the previous time $t-1$ and the transition likelihood $J_{t,i,j}^{t-1}$, using Equation (14). For each switching state i, an optimal previous state j, is kept in the state transition record $\psi_{t-1,i}$ using Equation (15). Consequently, we can obtain a set of N_m best filtered LDS states and state covariances at time t:

$$\hat{X}_{t,i}^{t} = \hat{X}_{t,i,\psi_{t-1,i}}^{t}, \quad (22)$$

$$\Sigma_{t,i}^{t} = \Sigma_{t,i,\psi_{t-1,i}}^{t}. \quad (23)$$

Once all the observations $Y_{1:T}$ have been processed, the maximum likelihood J_T^* and the best final switching state S_T^* can be computed as:

$$J_T^* = \max_i J_{T,i}, \quad (24)$$

$$S_T^* = \arg\max_i J_{T,i}. \quad (25)$$

Accordingly, the best switching state sequence $S_{1:T}^*$ can be obtained by backtracking the state transition record $\psi_{t-1,i}$:

$$S_t^* = \psi_{t,S_{t+1}^*}, \quad 1 \le t \le T-1. \quad (26)$$

The approximate Viterbi inference algorithm is summarized in Algorithm 1. The complexity of this algorithm is $O({N_m}^2 T)$, where N_m is number of LDS models and T is the length of an observation sequence.

Algorithm 1 Approximate Viterbi Inference Algorithm (E-step)

Input: $Y_{1:T} = \{Y_1, Y_2, \ldots, Y_T\}$ – a sequence of observed measurements
Θ – the current model parameters
Output: the optimal switching state sequence $S^*_{1:T}$
Procedure:

(1) Initialize the LDS state estimates $\hat{X}^1_{1,i}$ and $\Sigma^1_{1,i}$.
(2) Iterate while $2 \le t \le T$
  for $i = 1 : N_m$
    for $j = 1 : N_m$
      Predict and filter the LDS state estimates $\hat{X}^{t-1}_{t,i,j}$, $\Sigma^{t-1}_{t,i,j}$, $\hat{X}^t_{t,i,j}$ and $\Sigma^t_{t,i,j}$ using Equation (16) (17) (19) (20).
      Compute the transition likelihood $J^{t-1}_{t,i,j}$ using Equation (18).
    end for
    Find the maximum likelihood $J_{t,i}$ and the state transition record $\psi_{t-1,i}$ using Equation (14) (15).
    Compute the LDS state estimates $\hat{X}^t_{t,i}$ and $\Sigma^t_{t,i}$ using Equation (22) (23).
  end for
(3) Compute the maximum likelihood J^*_T and the best final switching state S^*_T using Equation (24) (25).
(4) Backtrack the best switching state sequence $S^*_{1:T}$ using Equation (26).

Based on the optimal switching state sequence, $S^*_{1:T}$, the sufficient statistics about the switching model are simply computed as $\langle S_t \rangle = \delta(i = S^*_t)$, $\langle S_t S^{'}_{t-1} \rangle = \delta(i = S^*_t)\delta(j = S^*_{t-1})$. Also, the sufficient statistics for the LDS models can be obtained using Rauch-Tung-Streiber (RTS) smoothing [Anderson and Moore 1979; Ghahramani 1998]. For example,

$$\langle X_t, S_t(i) \rangle = \hat{X}^T_{t,S^*_t}\, \delta(i = S^*_t), \tag{27}$$

$$\langle X_t X^{'}_t, S_t(i) \rangle = \left(X^T_{t,S^*_t} X^{T\ '}_{t,S^*_t} + \Sigma^T_{t,S^*_t} \right) \delta(i = S^*_t), \tag{28}$$

$$\langle X_t X^{'}_{t-1}, S_t(i,j) \rangle = \left(X^T_{t,S^*_t} X^{T\ '}_{t-1,S^*_{t-1}} + \Sigma^T_{t,t-1} \right) \delta(i = S^*_t)\delta(j = S^*_{t-1}). \tag{29}$$

5.3.2 *M-step: Maximum Likelihood Learning.* Given the sufficient statistics calculated from the inference in the E-step, the model parameters of the SLDS model $\Theta = \{A_i, Q_i, C, R, M, \pi\}$ can be updated using the maximum likelihood method [Anderson and Moore 1979; Ghahramani 1998]. Each parameter is re-estimated by taking the corresponding partial derivative of the expected log-likelihood, setting it to zero, and solving. The parameter equations are shown as follows:

$$A^{new}_i = \left(\sum_{t=2}^{T} \langle X_t X^{'}_{t-1}, S_t(i,j) \rangle \right) \left(\sum_{t=2}^{T} \langle X_{t-1} X^{'}_{t-1}, S_t(j) \rangle \right)^{-1}, \tag{30}$$

$$Q^{new}_i = \frac{1}{T-1} \left(\sum_{t=2}^{T} \langle X_t X^{'}_t, S_t(i) \rangle - A^{new}_i \sum_{t=2}^{T} \langle X_{t-1} X^{'}_t, S_t(i,j) \rangle \right), \tag{31}$$

$$C^{new} = \left(\sum_{t=1}^{T} Y_t \langle X_t^{'} \rangle\right) \left(\sum_{t=1}^{T} \langle X_t X_t^{'} \rangle\right)^{-1}, \tag{32}$$

$$R^{new} = \frac{1}{T} \sum_{t=1}^{T} (Y_t Y_t^{'} - C^{new} \langle X_t \rangle Y_t^{'}), \tag{33}$$

$$M^{new} = \left(\sum_{t=2}^{T} \langle S_t S_{t-1}^{'} \rangle\right) diag \left(\sum_{t=2}^{T} \langle S_t \rangle\right)^{-1}, \tag{34}$$

$$\pi^{new} = \langle S_1 \rangle. \tag{35}$$

After the learning process, each goal G_i, is associated with a set of model parameters Θ, which can be used to perform goal recognition in the online phase.

5.4 Online Goal Recognition Phase

In the online phase, given a new sequence of observations, $Y_1, Y_2, \ldots, Y_t$, obtained up to time t, we can infer the most likely goal G^*, as follows:

$$G^* = \arg\max_{G_k} P(G_k | Y_1, Y_2, \ldots, Y_t) = \arg\max_{G_k} P(G_k | Y_{1:t}). \tag{36}$$

By applying Bayes' Rule, this equation becomes:

$$G^* = \arg\max_{G_k} \frac{P(Y_{1:t}|G_k)P(G_k)}{P(Y_{1:t})} = \arg\max_{G_k} P(Y_{1:t}|G_k)P(G_k), \tag{37}$$

where the term $P(Y_{1:t})$ is a constant and can be dropped when performing the comparison. $P(Y_{1:t}|G_k)$ denotes the likelihood of each goal G_k, in generating the sequence of observations, $Y_{1:t}$. For each goal G_k, the calculation of $P(Y_{1:t}|G_k)$ is equivalent to computing $P(Y_{1:t}|\Theta_k)$, the likelihood of generating $Y_{1:t}$, given the corresponding model parameters Θ_k, learned from the training data. This is done by applying the inference algorithm given in Algorithm 1. At each time t, we compute the likelihood of segmentation $J_t^*(k)$ for each goal G_k in step (3). The goal G_k with the maximum likelihood is predicted as the recognized goal G^*.

6. EXPERIMENTAL RESULTS

In the previous sections, we have presented two approaches for activity recognition. The location-based activity recognition (LAR) algorithm, presented in Section 4, relies on a location-based sensor model to infer locations and then goals. The segmentation-based activity recognition algorithm (SAR), presented in Section 5, applies a segmentation-based learning algorithm to perform goal recognition. However, which approach is more accurate for activity recognition? Which one is less reliant on manual calibration? Which method is more suitable for real-time computation? In this section, we answer these questions by carrying out experiments on a real data set collected from a WLAN environment.

Table II. Goal Table

Goal #	Name	Goal #	Name
G_1	*Office-to-Entrance1*	G_2	*Office-to-Entrance2*
G_3	*Office-to-Entrance3*	G_4	*Entrance3-to-Office*
G_5	*Office-to-Print-in-Room1*	G_6	*Office-to-Seminar-in-Room1*
G_7	*Office-to-Print-in-Room2*	G_8	*Office-to-Seminar-in-Room2*
G_9	*Entrance3-to-Print-in-Room2*	G_{10}	*Entrance3-to-Seminar-in-Room2*
G_{11}	*Entrance3-to-Entrance2*	G_{12}	*Office-to-Entrance3*
G_{13}	*Entrance3-to-Entrance1*	G_{14}	*Entrance1-to-Seminar-in-Room2*
G_{15}	*Entrance2-to-Seminar-in-Room2*	G_{16}	*Entrance2-to-Office*
G_{17}	*Entrance1-to-Office*	G_{18}	*Entrance1-to-Seminar-in-Room1*
G_{19}	*Entrance2-to-Seminar-in-Room1*		

6.1 Experimental Setting

Our experimental test-bed was set up in the office area of the Department of Computer Science and Engineering at the Hong Kong University of Science and Technology. The building is equipped with an IEEE 802.11b wireless network in the 2.4 GHz frequency band. The layout of the floor is shown in Figure 1. This area has a dimension of 64×50 meters. In this environment, a user with a mobile device can carry out different activities in the three main areas (Office, Room1, and Room2) and seven hallways. Room1 and Room2 both provide facilities for printing services and holding seminars. The mobile device can periodically record signal-strength measurements propagated from the APs. A user's behavior trace is thus represented as a sequence of signal-strength measurements recording his movements in the environment.

We modeled 19 different goals of a professor's activities in this environment, such as *Office-to-Entrance1*, *Office-to-Seminar-in-Room1*, and *Office-to-Print-in-Room2*. All the goals are listed in Table II. We collected 864 traces using the device driver and API we developed, which operated on a LINKSYS wireless card. We used eight APs that can be detected in the environment. Each trace is therefore an eight-dimensional time series of signal-strength measurements. In addition, each trace was labeled with its intended goal by hand for evaluation purposes.

In our experiments, we modeled each activity as a multistate SLDS model based on the SAR algorithm. For each SLDS model, we set the observation matrix C to be an identity matrix I, and estimated the other parameters A_i, Q_i, R, M, and π using the EM-learning framework with the approximate Viterbi inference.

6.2 Evaluation Criteria

In our experiments, we use the following four criteria to evaluate the performance of the two algorithms.

(1) *Calibration Effort:* this criterion is measured in terms of the number of training samples calibrated for each location. Since collecting and labeling each signal-strength value takes approximately the same amount of

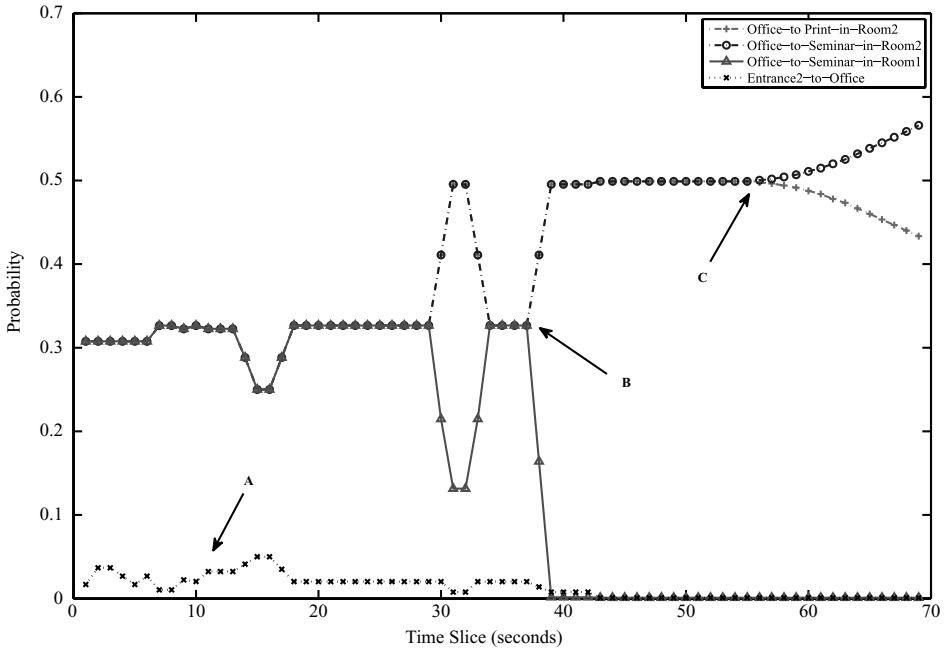

Fig. 8. Illustration of the activity recognition process.

time, the number of samples is proportional to the person-hours used for calibration.

(2) *Recognition Accuracy:* this criterion measures how accurate an algorithm is in recognizing a goal from a user's trace. For a certain goal, recognition accuracy is defined as the number of correct recognitions divided by the total number of recognitions that are performed. When dealing with multiple goals, the averaged recognition accuracy is reported.

(3) *Inference Efficiency:* inference efficiency is measured in terms of the average processing time for each observation during the online activity recognition.

(4) *Robustness:* this criterion measures how robust an algorithm is with respect to the observation noise contained in the user traces.

In the following, we directly used our collected real data to evaluate the first three criteria. In contrast, in order to evaluate the robustness, we added Gaussian noise to our real data after it was collected so as to simulate the variations of signal-strength values in a dynamically changing WLAN environment.

6.3 Illustration of the Activity Recognition Process

We now use an example to illustrate the process of activity recognition in the online phase. Figure 8 shows the change of probabilities for one user trace to achieve the goal *Office-to-Seminar-in-Room2* against the three other goals among the 19 goals. As shown in the figure, at the beginning, the probabilities of the three goals, *Office-to-Seminar-in-Room2*, *Office-to-Print-in-Room2*, and

Office-to-Seminar-in-Room1, are approximately equal. This is because, for the three goals, the user sets off from the same starting point, for example, *Office*, which thus leads the discovered motion patterns in the signal space to be similar to each other. As time moves on, the probabilities of *Office-to-Seminar-in-Room2* and *Office-to-Print-in-Room2* increase steadily when the user begins to take the action *Walk-in-HW4* at time point B. In contrast, the probability of achieving the goal Office-to-Seminar-in-Room1 decreases remarkably because the motion patterns associated with the action *Walk-in-HW4* cannot be fit well by its corresponding model parameters. After the user takes the action, *Enter-Room2*, the two goals *Office-to-Seminar-in-Room2* and *Office-to-Print-in-Room2*, cannot be differentiated right away because they may still share a common subsequence of signal-strength measurements. Finally, after time point C, the probability of *Office-to-Seminar-in-Rooms* remains the highest when the user starts to attend the seminar.

6.4 Effect of the Number of LDS Models

We investigate the effect of the number of LDS models, N_m, on the performance of the SAR algorithm in this section. As discussed in Section 5.2, N_m is an important parameter of the SAR algorithm because it affects the parameter estimation of the LDS models fit to signal sequences. Therefore, an optimal value of N_m should be determined properly so that the global nonlinear dynamics involved in user traces can be captured.

We adopted an iterative process to determine the optimal value of N_m for each goal in the learning phase. Specially, we varied the value of N_m from 1 to N_{max}, a prespecified maximum number of LDS models. For a given value of N_m, we computed the log-likelihood J_T^* of the observation sequences generated by the fit model, using Equation (24). For each goal, we found the optimal value of N_m such that the corresponding log-likelihood is maximized. In our experiments, N_{max} was set to 16 for all the goals, and with the iterative process, the optimal value of N_m was automatically determined for each goal through experiments.

Figure 9 illustrates the change of log-likelihoods for two goals, $G_1 =$ *Office-to-Entrance1* and $G_2 =$ *Office-to-Entrance2*, with respect to different numbers of LDS models, N_m. We can see from the figure that the optimal values of N_m are 8 and 12, for G_1 and G_2, respectively. This is because, for the two goals, the maximum log-likelihood of signal sequences generated by the corresponding model can be achieved at these two points. We can also observe that, when N_m is either too large or too small, the log-likelihood of signal sequences decreases remarkably. This occurs because, when N_m is too small or too large, the model parameters cannot be accurately estimated due to the effect of overfitting or underfitting. As a result, the performance of the SAR algorithm will also significantly degrade.

6.5 Overall Evaluation

We performed experiments to evaluate the performance of the LAR algorithm and the SAR algorithm over a total of 864 traces. For validity of experimental

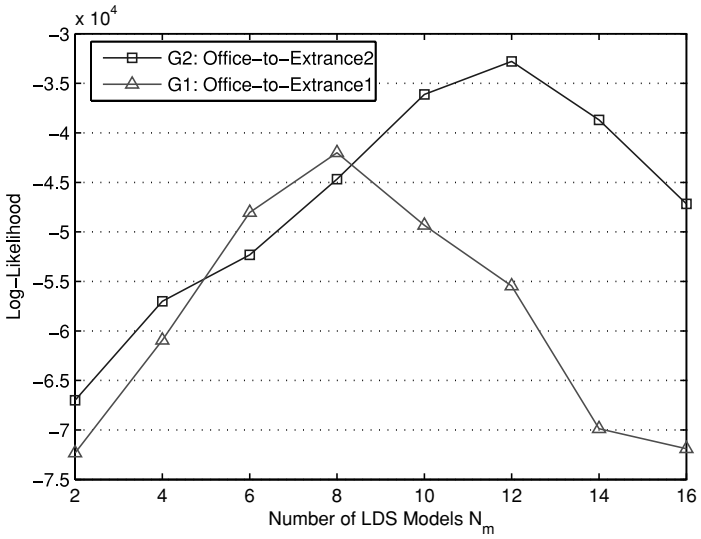

Fig. 9. Log-likelihood vs. number of LDS models, N_m.

results, we ran all the experiments using a three-fold cross-validation and reported the average results over the 19 goals.

6.5.1 *Calibration Effort.* We first compare the LAR algorithm and the SAR algorithm with respect to the calibration effort used to train the activity recognition model. Figure 10 shows the recognition accuracy of the two algorithms by varying the number of labeled training samples for each location. For the LAR algorithm, these labeled training samples are used to build the location-based sensor model. We varied the number of training samples for each location from 0 to 100. Similar to Figure 3, we ran this experiment through a three-fold cross validation over 864 user traces; 576 user traces were used for training and 288 for testing in each run. Each value plotted in the figure is averaged over three runs.

Let us look at Figure 10 in detail. When there are no labeled data with locations, the recognition accuracy of the LAR algorithm is 33.7%. As the number of labeled training samples for each location increases, the LAR algorithm can achieve higher recognition accuracy. In particular, when the number of training samples increases to 100, the recognition accuracy increases to 92.7%. In contrast, the SAR algorithm performs segmentation directly for activity recognition in the signal sequences. Thus, it does not require the labeled training data with locations. Accordingly, the recognition accuracy of the SAR algorithm remains 91.0%, which does not vary with respect to the number of training data with locations. In this experiment, we can learn about 50 motion patterns from 576 training traces by applying our learning algorithm.

In summary, the recognition accuracy of the LAR algorithm depends to a large extent on the number of training samples calibrated for each location. When there are not enough training samples used to train the location-based sensor model, the performance of the LAR algorithm is quite poor. However,

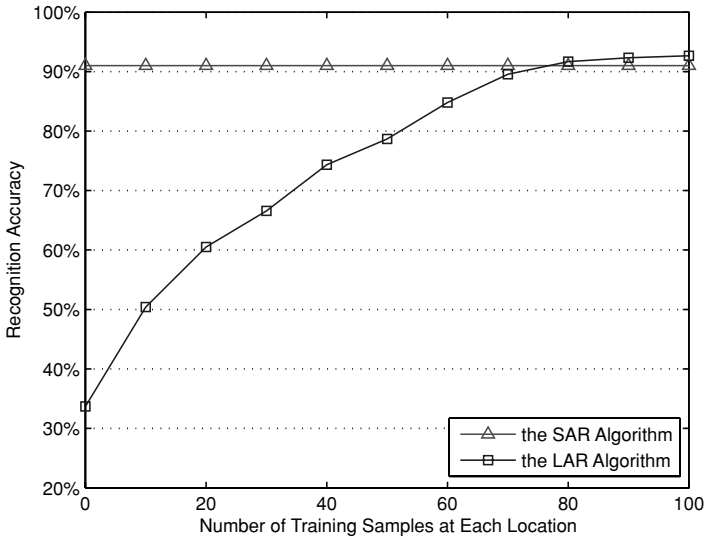

Fig. 10. Recognition accuracy vs. number of labeled training samples for each location.

the SAR algorithm can achieve comparably high recognition accuracy without the labeled training data for each location.

6.5.2 *Recognition Accuracy.* We then compare the recognition accuracy of the LAR algorithm and the SAR algorithm by varying the number of training traces. We varied the number of training traces from 96 to 576, and used 288 traces for testing. The LAR algorithm requires two data sources: the training traces, and the calibrated data with locations. The latter is used for building a sensor model; the number of training samples at each location for building a sensor model was set to be 100. In contrast, the SAR algorithm only requires the training traces as an input.

Figure 11 shows the recognition accuracy of the two algorithms with respect to different numbers of training traces. As shown in the figure, when the number of training traces increases, the recognition accuracy of the two algorithms increases steadily. This occurs because, the model parameters can be estimated more accurately when more training data are used. As a whole, the performance of the SAR algorithm without building sensor models is comparable to that of the LAR algorithm. In particular, when the training data are sparse, the SAR algorithm can even outperform the LAR algorithm. This is because the LAR algorithm employs a hierarchical model to infer a user's locations, actions and goals based on the sensor model. In this model, a large number of parameters need to be estimated, and as a result, an accurate estimation cannot be obtained without sufficient training data. The SAR algorithm, in contrast, relies on the characteristics of signals themselves to generate motion patterns, which reduces the number of parameters that need to be estimated. Therefore, less training data are required in the learning process. In summary, the SAR

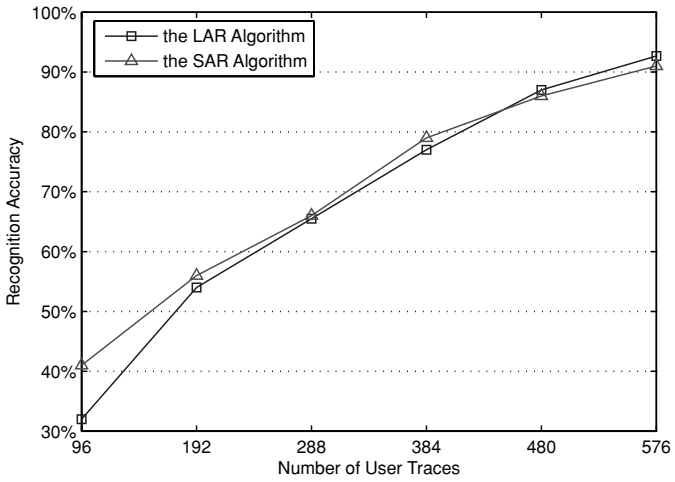

Fig. 11. Recognition accuracy vs. number of training traces.

algorithm can achieve comparable recognition accuracy to the LAR algorithm while effectively reducing the calibration effort.

6.5.3 *Inference Efficiency.* We now analyze the online inference efficiency of the two algorithms. In theory, the complexity of inference for the LAR algorithm is $O(K^w T)$, where K is the maximum number of possible values that each discrete hidden state can take, and w is the maximum size of cliques constructed using the junction tree algorithm. On the other hand, the complexity of inference for the SAR algorithm is $O({N_m}^2 T)$, where N_m is number of LDS models. Therefore, we can conclude that, for both algorithms, the time complexity of online inference is linear with the length of an observation sequence T. In addition, we also empirically compare the inference efficiency of the two algorithms with respect to the average processing time for the observations. The results are reported in Table III. We can see from the table that the average processing time for the observations is 0.30 and 0.20 seconds, for the LAR algorithm and the SAR algorithm, respectively. The SAR algorithm can be seen to be more efficient than the LAR algorithm because, in our experiments, the state space for the LAR algorithm is much larger than the that of the SAR algorithm. However, experiments show that, using both algorithms, the activities can be recognized in real time.

6.5.4 *Robustness.* Finally, we compare the robustness of the LAR algorithm and the SAR algorithm with respect to the observation noise contained in the user traces. As pointed out in Haeberlen et al. [2004] and Yin et al. [2005b], the signal-strength values may vary a lot over time in a dynamic WLAN environment. This causes the received signal-strength values contained in the traces to deviate from the calibration data used for building a sensor model. In this experiment, we simulated the variations of signal strength by adding different levels of Gaussian noise to the signal sequences. We

Table III. Average Processing Time for Each Observation

Method	Average Processing Time (seconds)
The LAR algorithm	0.30
The SAR algorithm	0.20

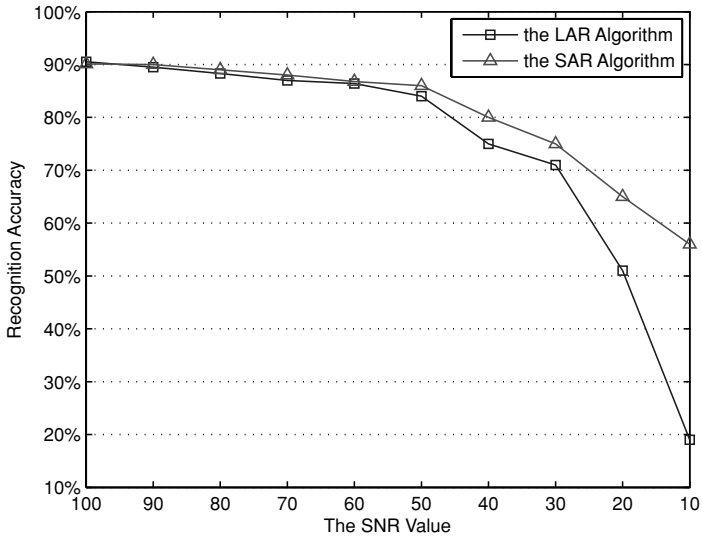

Fig. 12. Recognition accuracy vs. the SNR value in signal sequences.

varied the relative signal-to-noise ratio (SNR, which is the ratio of variance of signal to variance of noise) of signal sequences from 100 to 10. The smaller the SNR value, the higher level of the Gaussian noise added into the signal sequences.

Figure 12 shows the recognition accuracy of the two algorithms with respect to different SNR values. For the LAR algorithm, the number of training samples at each grid was set to 100. We can see from the figure that, as the SNR value of signal sequences decreases, that is, when the noise level increases, the performance of the two algorithms decreases. This occurs because when more noise is involved in the signal sequences, the model parameters for each activity cannot be accurately estimated from the training data. As a result, the recognition accuracy of the two algorithms decreases. However, when the SNR value is small, the SAR algorithm can outperform the LAR algorithm. This indicates that, when more noise is involved in the signal sequences, for the LAR algorithm, the sensor model cannot accurately estimate the locations from signal-strength values. This subsequently degrades the performance of activity recognition remarkably. In contrast, the SAR algorithm can still capture the significant trends contained in the signal segments. Therefore, the SAR algorithm is more robust to the observation noise when the signal strength varies over time in a dynamic WLAN environment.

6.6 Experimental Summary

Based on the experimental results presented above, we now summarize the advantages of the SAR algorithm as follows:

(1) Compared with the LAR algorithm, the SAR algorithm can effectively reduce the calibration effort while achieving comparable recognition accuracy.
(2) The SAR algorithm is more robust to the observation noise involved in signal sequences, when the signal strength changes in a dynamic WLAN environment.

7. CONCLUSIONS AND FUTURE WORK

In this article, we have proposed a novel activity-recognition algorithm that segments low-level signal sequences with a goal-based probabilistic model. In our approach, we apply a probabilistic segmentation model, in which each segment of signals is represented as an LDS model and the transitions among signal segments as a Markov process conditioned on goals. Our EM learning algorithm can simultaneously learn the motion patterns and their transition models depending on different goals, which in turn can be used to accurately recognize activities in an online phase. Our experimental results on a real data set demonstrate that our proposed SAR algorithm is both accurate and robust for activity recognition with less calibration effort.

In the future, we plan to extend our work in several directions. First, although we adopt the approximate Viterbi inference algorithm as a subroutine to learn the SLDS model for activity recognition, we also wish to test the feasibility of other approximate inference algorithms, such as GPB2 [Pavlović et al. 2000] and MCMC [Oh et al. 2005], for learning the activity recognition models.

Second, although we evaluate the performance of the SAR algorithm based on a WLAN environment, our SAR method has been developed as a general probabilistic framework to perform activity recognition in a wide range of sensor-network environments. Therefore, we will use other motion sensors, such as RFID, accelerometer, and infrared, to test the effectiveness of the SAR algorithm. Using the additional information provided by these sensors, we believe that we will be able to perform fine-grained activity recognition [Tapia et al. 2004; Wren and Tapia 2006].

Third, we demonstrate that the SAR algorithm can achieve recognition accuracy comparable to the LAR algorithm, even without requiring the calibrated data for locations. However, we believe our SAR algorithm can still benefit from the useful contextual information, such as the areas and locations, in the environments. Therefore, we wish to further explore how to apply semi-supervised learning [Zhu 2005] to bootstrap the performance of the SAR algorithm, by using a small amount of training data labeled with locations.

Fourth, using our SAR algorithm, activity recognition is performed at the mobile clients in a centralized manner. To scale up activity recognition, we would like to investigate how to efficiently distribute the computation of the SAR algorithm across a sensor network. This can be achieved by using a hierarchical organization of inferences, in which simple motion patterns inferred

locally at sensor nodes can be used to reason about more macroscopic behaviors [Lymberopoulos et al. 2006].

Finally, we plan to continue in this direction in reusing the motion patterns that are obtained in this analysis. One application is to use them for the task of planning, and another is to recognize abnormal activities performed by unknown agents for security monitoring applications.

ACKNOWLEDGMENT

We thank the anonymous referees for their valuable comments.

REFERENCES

ALBRECHT, D., ZUKERMAN, I., AND NICHOLSON, A. 1998. Bayesian models for keyhole plan recognition in an adventure game. *User Model. User-Adapt. Interac. 8*, 1–2, 5–47.

ANDERSON, B. D. O. AND MOORE, J. B. 1979. *Optimal Filtering*. Prentice-Hall, Englewood Cliffs, New Jersey.

BAHL, P., BALACHANDRAN, A., AND PADMANABHAN, V. 2000. Enhancements to the RADAR user location and tracking system. Tech. rep. MSR-TR-2000-12, Microsoft Research.

BAHL, P. AND PADMANABHAN, V. N. 2000. RADAR: An in-building RF-based user location and tracking system. In *Proceedings of the 19th Annual Joint Conference of the IEEE Computer and Communications Societies (INFOCOM)*. Tel-Aviv, Israel, 775–784.

BLAYLOCK, N. AND ALLEN, J. 2003. Corpus-based statistical goal recognition. In *Proceedings of the 8th International Joint Conference on Artificial Intelligence (IJCAI)*. Acapulco, Mexico, 1303–1308.

BREGLER, C. 1997. Learning and recognizing human dynamics in video sequences. In *Proceedings of the International Conference on Computer Vision and Pattern Recognition (CVPR)*. San Juan, Puerto Rico, 568–574.

BUI, H., PHUNG, D., AND VENKATESH, S. 2004. Hierarchical hidden Markov models with general state hierarchy. In *Proceedings of the 19th National Conference on Artificial Intelligence (AAAI)*. San Jose, CA, 324–329.

BUI, H., VENKATESH, S., AND WEST, G. 2002. Policy recognition in the abstract hidden Markov model. *J. Art. Intel. Res. 17*, 451–499.

CHAI, X. AND YANG, Q. 2005. Reducing calibration effort for location estimation using unlabeled samples. In *Proceedings of IEEE International Conference on Pervasive Computing and Communications (PerCom)*. Kauai Island, HW, 95–104.

CHARNIAK, E. AND GOLDMAN, R. 1993. A Bayesian model of plan recognition. *AI 64*, 53–79.

CHIU, B., KEOGH, E., AND LONARDI, S. 2003. Probabilistic discovery of time series motifs. In *Proceedings of the International Conference on Knowledge Discovery and Data Mining (KDD)*. Washington, DC, 493–498.

CZIELNIAK, G., BENNEWITZ, M., AND BURGARD, W. 2003. Where is . . . ? learning and utilizing motion patterns of persons with mobile robots. In *Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI)*. Acapulco, Mexico, 909–914.

DEMPSTER, A. P., LAIRD, N. M., AND RUBIN, D. B. 1977. Maximum likelihood from incomplete data via EM algorithm. *J. Royal Statis. Soc. Series B 39*, 1–38.

ENGE, P. AND MISRA, P. 1999. Special issue on GPS: The global positioning system. *Proc. IEEE*, 3–172.

FINE, S., SINGER, Y., AND TISHBY, N. 1998. The hierarchical hidden Markov model: Analysis and applications. *Mach. Learn. 32*, 1, 41–62.

FOX, D., HIGHTOWER, J., LIAO, L., AND SCHULZ, D. 2002. Bayesian filtering for location estimation. *IEEE Perv. Comput. 2*, 3, 24–33.

GHAHRAMANI, Z. 1998. Learning dynamic Bayesian networks. In *Adaptive Processing of Temporal Information, Lecture Notes in Artificial Intelligence*, C. L. Giles and M. Gori, Eds. Springer-Verlag, Berlin, 168–197.

GHAHRAMANI, Z. AND HINTON, G. E. 1998. Switching state-space models. Tech. Rep., 6 King's College Road, Toronto M5S 3H5, Canada.

GOLDMAN, R., GEIB, C., AND MILLER, C. 1999. A new model of plan recognition. In *Proceedings of the 15th Annual Conference on Uncertainty in Artificial Intelligence (UAI)*. Stockholm, Sweden, 245–254.

HAEBERLEN, A., FLANNERY, E., LADD, A., RUDYS, A., AND KAVRAKI, D. W. L. 2004. Practical robust localization over large-scale 802.11 wireless networks. In *Proceedings of the 10th ACM International Conference on Mobile Computing and Networking (MobiCom)*. Philadelphia, PA, 70–84.

HAN, K. AND VELOSO, M. 2000. Automated robot behavior recognition applied to robotic soccer. In *Robotics Research: the 9th International Symposium*. Springer-Verlag, London, 199–204.

HIMBERG, J., KORPIAHO, K., MANNILA, H., TIKANMAKI, J., AND TOIVONEN, H. 2001. Time series segmentation for context recognition in mobile devices. In *Proceedings of the 1st International Conference on Data Mining (ICDM)*. San Jose, CA, 203–210.

KAUTZ, H. AND ALLEN, J. F. 1986. Generalized plan recognition. In *Proceedings of the 5th National Conference on Artificial Intelligence (AAAI)*. Philadelphia, PA, 32–37.

KEOGH, E. J., CHU, S., HART, D., AND PAZZANI, M. J. 2001. An online algorithm for segmenting time series. In *Proceedings of the International Conference on Data Mining (ICDM)*. San Jose, CA, 289–296.

LADD, A., BEKRIS, K., MARCEAU, G., RUDYS, A., KAVRAKI, L., AND WALLACH, D. 2002. Robotics-based location sensing using wireless Ethernet. In *Proceedings of the 8th Annual International Conference on Mobile Computing and Networking (MobiCom)*. Atlanta, GA, 227–238.

LARI, K. AND YOUNG, S. J. 1990. The estimation of stochastic context-free grammars using the inside-outside algorithm. *Comput. Speech Lang. 4*, 35–56.

LESH, N. AND ETZIONI, O. 1995. A sound and fast goal recognizer. In *Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI)*. Montreal, Canada, 1704–1710.

LI, Y., WANG, T., AND SHUM, H.-Y. 2002. Motion texture: a two-level statistical model for character motion synthesis. In *Proceedings of the International Conference on Computer Graphics and Interactive Techniques (SIGGRAPH)*. San Antonio, TX, 465–472.

LIAO, L., FOX, D., AND KAUTZ, H. 2004. Learning and inferring transportation routines. In *Proceedings of the 19th National Conference in Artificial Intelligence (AAAI)*. San Jose, CA, 348–353.

LOKE, S. 2005. Representing and reasoning with situations for context-aware pervasive computing: a logic programming perspective. *Know. Eng. Rev. 19*, 3, 213–233.

LYMBEROPOULOS, D., OGALE, A., SAVVIDES, A., AND ALOIMONOS, Y. 2006. A sensory grammar for inferring behaviors in sensor networks. In *Proceedings of the 5th International Conference on Information Processing in Sensor Networks (IPSN)*. Nashville, Tennessee, 251–259.

MINKA, T. 1998. Expectation-maximization as lower bound maximization. Tutorial. `http://research.microsoft.com/~minka/papers/em.html`.

MUHLENBROCK, M., BRDICZKA, O., SNOWDON, D., AND MEUNIER, J.-L. 2004. Learning to detect user activity and availability from a variety of sensor data. In *Proceedings of the 2nd IEEE Annual Conference on Pervasive Computing and Communications (PerCom)*. Orlando, Florida, 13–22.

MURPHY, K. 1998. Learning switching Kalman filter models. Tech. rep. TR 98–10, Compaq Cambridge Research Lab.

MURPHY, K. 2002. Dynamic Bayesian networks: Representation, inference and learning. Ph.D. thesis, University of California Berkeley.

NGUYEN, N., BUI, H., VENKATESH, S., AND WEST, G. 2003. Recognising and monitoring high-level behaviours in complex spatial environments. In *Proceedings of International Conference on Computer Vision and Pattern Recognition (CVPR)*. Madison, WI, 620–625.

OATES, T. 2002. PERUSE: An unsupervised algorithm for finding recurring patterns in time series. In *Proceedings of the International Conference on Data Mining (ICDM)*. Maebashi City, Japan, 330–337.

OH, S. M., REGH, J. M., BALCH, T., AND DELLAERT, F. 2005. Data-driven MCMC for learning and inference in switching linear dynamic systems. In *Proceedings of the 20th National Conference in Artificial Intelligence (AAAI)*. Pittsburgh, PA, 944–949.

PATTERSON, D., LIAO, L., FOX, L., AND KAUTZ, H. 2003. Inferring high-level behavior from low-level sensors. In *Proceedings of the 5th International Conference on Ubiquitous Computing (UbiComp)*. Seattle, WA, 73–89.

PAVLOVIĆ, V. AND REHG, J. M. 2000. Impact of dynamic model learning on classification of human motion. In *Proceedings of the International Conference on Computer Vision and Pattern Recognition (CVPR)*. Hilton Head Island, SC, 788–795.

PAVLOVIĆ, V., REHG, J. M., CHAM, T.-J., AND MURPHY, K. 1999. A dynamic Bayesian network approach to figure tracking using learned dynamic models. In *Proceedings of the 6th IEEE International Conference on Computer Vision (ICCV)*. Kerkyra, Corfu, Greece, 94–101.

PAVLOVIĆ, V., REHG, J. M., AND MACCORMICK, J. 2000. Learning switching linear models of human motion. In *Advances in Neural Information Processing Systems (NIPS)*. Vol. 13. MIT Press, Cambridge, MA, 981–987.

PEURSUM, P., BUI, H., VENKATESH, S., AND WEST, G. 2004. Human action segmentation via controlled use of missing data in hmms. In *Proceedings of the International Conference on Pattern Recoginition (ICPR)*. Cambridge, UK, 440–445.

PRIYANTHA, N. B., CHAKRABORTY, A., AND BALAKRISHNAN, H. 2000. The cricket location-support system. In *Proceedings of the 6th International Conference on Mobile Computing and Networking (MobiCom)*. Boston, MA, 32–43.

PYNADATH, D. V. AND WELLMAN, M. P. 2000. Probabilistic state-dependent grammars for plan recognition. In *Proceedings of the 16th Conference on Uncertainty in Artificial Intelligence (UAI)*. San Francisco, CA, 507–514.

ROOS, T., MYLLYMAKI, P., TIRRI, H., MISIKANGAS, P., AND SIEVANEN, J. 2002. A probabilistic approach to WLAN user location estimation. *Int. J. Wireless Inform. Netw. 9*, 3 (July), 155–164.

SMAILAGIC, A. AND KOGAN, D. 2002. Location sensing and privacy in a context aware computing environment. *IEEE Wirel. Comm. 9*, 5, 10–17.

TAPIA, E. M., INTILLE, S., AND LARSON, K. 2004. Activity recognition in the home using simple and ubiquitous sensors. In *Proceedings of the 2nd International Conference on Pervasive Computing (Pervasive)*. Vienna, Austria, 158–175.

TEKINAY, S. 1998. Special issue on wireless geolocation systems and services. *IEEE Comm. Mag. 87*, 1 (April).

WANT, R., HOPPER, A., FALCO, V., AND GIBBONS, J. 1992. The active badge location system. *ACM Trans. Inform. Sys. 10*, 1 (January), 91–102.

WREN, C. R. AND TAPIA, E. M. 2006. Toward scalable activity recognition for sensor networks. In *Proceedings of the 2nd International workshop in Location and Context-Awareness (LoCA)*. Vol. 3987. Dublin, Ireland, 168–185.

YIN, J., CHAI, X., AND YANG, Q. 2004. High-level goal recognition in a wireless LAN. In *Proceedings of the 19th National Conference in Artificial Intelligence (AAAI)*. San Jose, CA, 578–584.

YIN, J., SHEN, D., YANG, Q., AND LI, Z.-N. 2005a. Activity recognition through goal-based segmentation. In *Proceedings of the 20th National Conference on Artificial Intelligence (AAAI)*. Pittsburgh, PA, 28–33.

YIN, J., YANG, Q., AND NI, L. M. 2005b. Adaptive temporal radio maps for location estimation. In *Proceedings of IEEE International Conference on Pervasive Computing and Communications (PerCom)*. Kauai Island, HW, 85–94.

YOUSSEF, M. AND AGRAWALA, A. 2004. Handling samples correlation in the horus system. In *Proceedings of the 23rd IEEE Conference on Computer Communications and Networking (INFOCOM)*. Hong Kong, China, 1023–1031.

YOUSSEF, M., AGRAWALA, A., AND SHANKAR, U. 2003. WLAN location determination via clustering and probability distributions. In *Proceedings of IEEE International Conference on Pervasive Computing and Communications (PerCom)*. Dallas, TX, 143–150.

ZAKI, M. J. 2001. SPADE: An efficient algorithm for mining frequent sequences. *Machine Learn. 42*, 1/2, 31–60.

ZHU, X. 2005. Semi-supervised learning literature survey. Tech. rep. 1530, Computer Sciences, University of Wisconsin-Madison.

Received April 2007; revised October 2007; accepted November 2007