A Location-Based Indoor Recommendation System Using A Hidden Markov Model

by

Yusheng Ding

A thesis submitted to the Graduate Faculty of
Auburn University
in partial fulfillment of the
requirements for the Degree of
Master of Science

Auburn, Alabama
May 07, 2016

Keywords: Recommendation System, RFID, Location-based services, Hidden Markov Model

Abstract

Recommendation systems play an important role in both the industrial and the academic worlds because they can combine theoretical study and market use together perfectly. Through improving user experience, recommendation systems bring considerable profits for businesses. However, most recommendation system applications are on the Internet. Due to the limitation of data-collecting techniques, economic costs, customer privacy protections and other factors, in offline retail stores, usually the same advertisement will be pushed to every customer with no differentiation. Advertisements are usually chosen only based on the selling rates of goods, which cannot deliver advertisements to customers precisely. With the development of RFID technology, a way of tracking customers' movements and purchasing behaviours with low costs and privacy protections is provided. Hidden Markov model (HMM) is a statistical Markov model in which the system being modeled is assumed to be a Markov process with unobserved (hidden) states. Therefore, it performs well when predicting results for things which patterns are hard to find. With only sequences of observations, HMM is able to predict the next emission. In this paper, a recommendation system that recommends goods for customers in a grocery store is proposed and implemented. To simplify the problem, our research considers zones in a grocery store that are detected and strictly separated by RFID readers and every shopping cart's movement stands for a customer's movement. Our system takes location information as well as customer movements into consideration as a parameter (state) for training the HMM model. Every one of our designed algorithms is compared with probability selection method, and we prove that by adding location and customer movements parameters, our solution has a higher hit rate for recommending the right goods for customers.

Acknowledgments

I owe my gratitude to all those people who have made this thesis possible and because of whom my graduate experience has been one that I will cherish forever.

Firstly, I would like to express my sincere gratitude to my advisor Prof. Wei-Shinn Ku for the continuous support of my Masters study and related research, for his patience, motivation, and immense knowledge. His guidance helped me in all the time of research and writing of this thesis. I could not have imagined having a better advisor and mentor for my Masters study.

Besides my advisor, I would like to thank the rest of my thesis committee: Prof. Xiao Qin and Prof. N. Hari Narayanan, for their insightful comments and encouragement.

My sincere thanks also goes to Dr. Senthilkumar CP, who provided me an opportunity to join his team as an intern, and who gave access to the laboratory and research facilities. Without their precious support it would not be possible to conduct this research. I also want to thank Dr. Jacqueline Hundley; I feel really grateful for have been working with her as a teaching assistant for two and a half years. Dr. Hundley gives me lots of advice and guidance both on how to be a good a teaching assistant and on my life path. She has always been very helpful for students and everybody around her.

I thank my fellow labmates: Jeff Wang, Liang Tang, Zhitao Gong, Ting Shen, Wenlu Wang, Ya Chi Kuo and Swagata Mukherjee for the stimulating discussions, for the sleepless nights we were working together before deadlines, and for all the fun we have had in the last few years. In particular, I am grateful to Prof. Yunpeng Xiao for guiding me in the first steps of research.

Many friends have helped me stay sane through these difficult years. Their support and care helped me overcome setbacks and stay focused on my graduate study. Here I want to thank my friends Bichong Chen, Shu Su, Xuerui Chen, Song Gao, Yuanqi Chen, Chunzhang Mo, Qi Zhu, Qian Zhang. I greatly value their friendship and I deeply appreciate their belief in me.

Most importantly, none of this would have been possible without the love and patience of my family. My immediate family to whom this thesis is dedicated, has been a constant source of love, concern, support and strength all these years. I would like to express my heart-felt gratitude to my parents and to my dearest wife Yawei Chen for providing me with unfailing support and continuous encouragement throughout my years of study and through the process of researching and writing this thesis. This accomplishment would not have been possible without them.

Table of Contents

List of Figures

List of Tables

Chapter 1
Introduction

Recommendation systems, which aim at predicting the "rating" or "preference" that a user would give to an item [19, 11], have been applied in a variety of applications in many fields and become extremely common in recent years. The most popular ones are probably movies, music, news, books, research articles, search queries, social tags, and products in general. However, there are also recommendation systems for experts, jokes, restaurants, financial services [8], life insurance, persons (online dating), and Twitter followers [12].

It's easy to understand that most of the Internet companies have applied recommendation systems in their products since recommendation systems do provide a better user experience and help those companies gain more profits through the process. However, it's hard to find a recommendation system using real-time data in a retail environment, such as a supermarket. According to W. Wade [22], it will be a trend for marketing people to exploit more location information, but due to the reasons that it's hard to collect users data because of customers' privacy protection and also indoor spaces are large and complex, people are not using recommendation system for real-time retail stores. However, the fact is that people spend a significant amount of time in indoor spaces. Consequently, the demand for launching indoor spatial queries is growing (e.g., find a person's location). Most of the time, people are active in some places frequently such as office buildings, schools, subway systems and many other structures. This makes it possible to predict people's behavior according to people's position records.

Most of the existing outdoor spatial query techniques are hard to realize in indoor situations. Most of them use GPS signals or mobile networks to find a position. However, it is often impossible to acquire indoor user locations by these methods. Furthermore, indoor spaces are far more complex than outdoor spaces. Object movements are limited by the walls, doors, hallways and so on.

Radio Frequency Identification (RFID) technologies are often employed for indoor tracking. RFID readers, which can automatically identify RFID tags, are often deployed in critical locations. When an object with a tag passes a reader's detection range, the reader will recognize it and generate a record in the database. However, the RFID raw data is inherently unreliable. Because of the high cost and privacy concerns, RFID readers cannot cover all indoor areas. Therefore, the RFID raw data cannot be used directly to evaluate indoor spatial queries.

When a person is often active in limited indoor space, his behavior always has some regularity. Therefore, his next destination can be predicted if we have the records of his paths and locations. However, the behavioral regularity cannot be found easily. A person's behavior is rather complex, which would be affected by many different factors. It's not simply statistics.

Our research considers a grocery store setting with a number of RFID readers deployed in critical locations, and merchandise, shopping carts are attached with RFID tags. Here we use shopping carts to indicate customers since it's not practical to attach RFID tags on customers. When a shopping cart with an RFID tag is in the detecting range of an RFID reader, it will be identified and this record will be saved into our database. The purpose of our system is to predict the user's destination and route in real time as well as answer indoor spatial queries to finally predict and recommend merchandise to customers based on his/her moving path and the historical data in our system.

Hidden Markov Model (HMM) is a statistical Markov model in which the system being modeled is assumed to be a Markov process with unobserved (hidden) states. HMM is often applied in temporal pattern recognition. In this paper, we propose Hidden Markov Model as the method for predicting the users' behavior from RFID raw data. The contributions of this study are as follows:

1. We apply HMM to analyze the object's behavior.

2. We propose HMM to predict the customer's destination and route as well as the customer's purchasing goods.

3. We propose and implement an indoor location-based recommendation system for a grocery store environment.

4. We evaluate our system through comparing our proposed algorithm with traditional statistical probabilities method.

The rest of the thesis first discusses related work in Chapter 2, preliminary in Chapter 3, and then describes our system design in Chapter 4. Chapter 5 describes the implementation and evaluation of our system. Chapter 6 presents our conclusions and describes future work.

Chapter 2

Related Work

In this chapter, some of studied literature related to recommendation systems and indoor location prediction measures are briefly presented.

2.1 Recommendation System

2.1.1 Common Recommendation System

Recommendation systems use the opinions of a community to help individuals in that community more effectively identify content of interest from a potentially overwhelming set of choices [25].

For online communities, recommendation systems can efficiently help users find topics and contents they are interested in, the effect is more obvious when for big online communities where there are a large numbers of the topics and contents in them [1].

Companies proving online shopping services like Amazon have admitted that with the exploiting of macro pattern in purchasing and browsing behaviors, user experience can be substantially enhanced [15]. Collaborative filtering is a common method used in recommendation systems trying to predict rating of a product for a particular user. The basic idea behind collaborative filtering is that similar users will vote on similar items similarly [18]. Thus, we can predict the vote of a particular user for some items when the similarity is determined between the users and items.

For social network services like Facebook, LinkedIn, Google+, recommendation systems are used for exploring people who potentially know each other and contents potentially be interesting for each particular user. Nearest Neighborhood Approach is the most commonly used algorithm [20]. By calculating the Pearson Correlation, we can find users in a social network who have the same or similar interests and tastes which defined those users' as neighbors for each other. Further, one particular user's preference can be predicted by calculating his or her preference data of top-N nearest neighbors through some certain techniques. In [21] Spertus et al. compared six distinct methods of calculating the similarity in a recommendation system for recommending online communities for Orkut social network users. As a result, they found that the best prediction results come from the cosine similarity measure among other measures.

2.1.2 Location-based Recommendation System

Location based recommendation systems are the recommendation systems take users geographic locations as parameters when calculating which contents (items, users, locations, etc.) will recommend to a particular user. In [4], Brunato et al. attempted to recommend websites to individual users according to the locations where they access the Web. In [26], Yang et al. proposed a location-aware recommender system which accommodates customers' shopping demands with location-dependent vendor offers and promotions.

However, none of those techniques mentioned in above two sections have considered the real-world location services, especially indoor scenarios which has to take location and space elements and restrictions into consideration.

2.2 Indoor Location Prediction

Some papers aim to predict the location by the assumption that the movement pattern exists.

Ghoutia et al. [9], proposed extreme learning machines for mobility prediction, though it is suited in the mobile ad hoc network, not same with our problem scenario.

To decrease severe received signal strength fluctuation, Chen et al. [6] used the model of Brownian motion to predict the position, built a Judgment system to choose the final results as the prediction or the estimating results defined by any fingerprint-based positioning algorithms. Finally, they improved the original results by the predictions. Though it has the element of prediction, it still does not provide a complete system for answering the query of prediction. Moreover, the model and aim are different from ours.

Kolodziej et al. [14, 13] used the movement history by hidden Markov model in their mobility prediction algorithm. Locations and transitions are employed as nodes in the multi-graph model. The edges are defined as the walkable paths. Therefore, the goal of this paper is to predict the activities of the (n+1) days in the time sequence by providing n-days of movement history. Thus their problem scenario is also different with ours.

Archibald et al. [3] used the rooms as the units, this is similar as us. They aim to let robots follow humans in the buildings. When robots are within line-of-sight of humans, they achieved the goal. Though there are walls in the buildings, the prediction is needed. This algorithm has two phases: pre-processing phase and run-time phase. In run-time phase, particle is calculated for define the shortest path that the humans go from the current location to the destination. We have the different model: Hidden Markov model to process this problem, and let the data be more useful. The other improvement is that we use the raw data from the readers on the corridor to improve the accuracy for the path. The paper assumes the shortest path as the result.

Chapter 3

Preliminary

In this chapter, brief introductions of Hidden Markov Model, HMM based user behavior prediction and brief introductions of Radio Frequency Identification (RFID) technology are provided.

3.1 Recommendation System

Recommendation systems, sometimes also referred as recommender systems (sometimes also replacing “system” with a synonym such as engine, platform, etc.) are a subclass of information filtering system that seek to predict the “rating” or “preference” that a user would give to an item [19, 10].

The typical outcome of a recommendation system is a list of recommendations in one of two ways: content-based filtering approaches and collaborative filtering approaches.

Collaborative filtering approaches usually first build a model from historical data of users’ behaviours, for example, goods are previously purchased or chosen and numerical ratings given to those items and similar decisions made by other users. The model will group the target predict user/item with other similar users/items then to predict items (rating for items) that the user may have an interest in [16].

Content-based filtering approaches usually analyze and utilize a series of discrete characteristic of the historical data of user/item in order to recommend additional items/users with similar properties [17].

In some recommendation systems, content-based filtering and collaborative filtering approaches are combined used. Figure 3.1 describes the overview of the relationship between those methods.

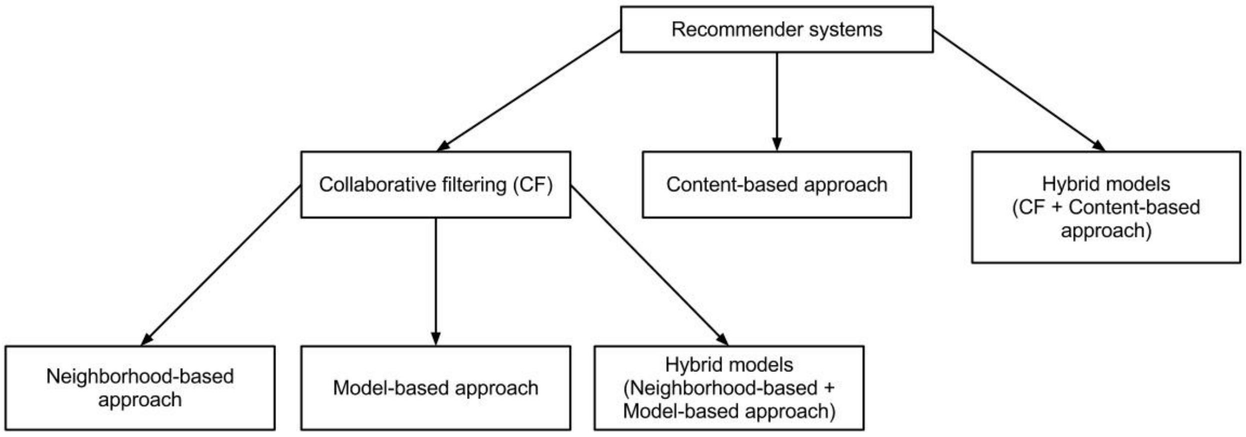

Recommender systems
Collaborative filtering (CF)
Content-based approach
Hybrid models (CF + Content-based approach)
Neighborhood-based approach
Model-based approach
Hybrid models (Neighborhood-based + Model-based approach)

Figure 3.1: Methods Used In Recommender Systems

3.1.1 Content-based filtering

Content-based filtering methods are based on a description of the item and a profile of the users preference [5]. Keywords describe user profiles and items are used in content-based recommendation system. And with those keywords, we can distinguish users or items in different groups, and then recommend items or users that are similar to those that a user liked or picked in the past (as well as is examining in the present). More particularly, by comparing various candidate items with items previously rated by the user and then recommended the best-matching users or items. This approach has its roots in information filtering and information retrieval research.

Item presentation algorithms are proposed and applied to abstract the items' features in recommendation system. A famous and wildly used algorithm is the vector space representation (also called tf-idf representation) [5].

Content-based recommendation mostly focuses on two types of information to create users and items profiles:

1. User preference model;

2. Historical data of Users' interaction with the system.

These methods use, for example a set of discrete attributes and features, items' and users' profiles characterizing the item and user in the system and then creating model based in a weighted vector of item features and users' profiles. For assigning the weight vector, a variety of techniques can be used. Simple approaches use the average values of the rated item vector while other sophisticated methods use machine learning techniques such as decision trees, Bayesian Classifiers, cluster analysis, and artificial neural networks in order to estimate the probability that a user is going to like or pick the item [25].

3.1.2 Collaborative filtering

Different from Content-based filtering, collaborative filtering is more focused on finding the relationship between users and items. In collaborative filtering, researching and analyzing a large amount of information of user behaviours, activities and preferences are an important part to archive the goal of predicting what the user will like based on their similarity to other users. By utilizing the relationship collaborative filtering has found between users (items), it is not necessary for a deep "understanding" of every attributes of the user (item) in the system. Many algorithms have been used in measuring user similarity or item similarity in recommender systems. For example, the k-nearest neighbor (k-NN) approach [20] and the Pearson Correlation as first implemented by Allen [2].

The assumption in collaborative filtering is users who agreed in the past will agree in the future, there is a very high chance for them to like (pick) the same, or at least similar kinds of items [23].

3.2 Radio Frequency Identification (RFID) Technology

Radio frequency identification (RFID) uses electromagnetic fields to automatically identyfy and track tags attached to objects [24]. Electronically stored information are embedded in RFID tags. A typical RFID system is made up of three components: tags, readers and the host computer system.

3.2.1 RFID Tags

An RFID tag is most referred to a tiny radio device or transponder, smart label, smart tag or radio barcode. A small, simple silicon microchip attached to a small flat aerial and mounted on a substrate is comprised in the tag. Then the whole device can be encapsulated in different materials dependent upon its intended usage. Figure 3.2.1 shows some examples of RIFD tags [7].

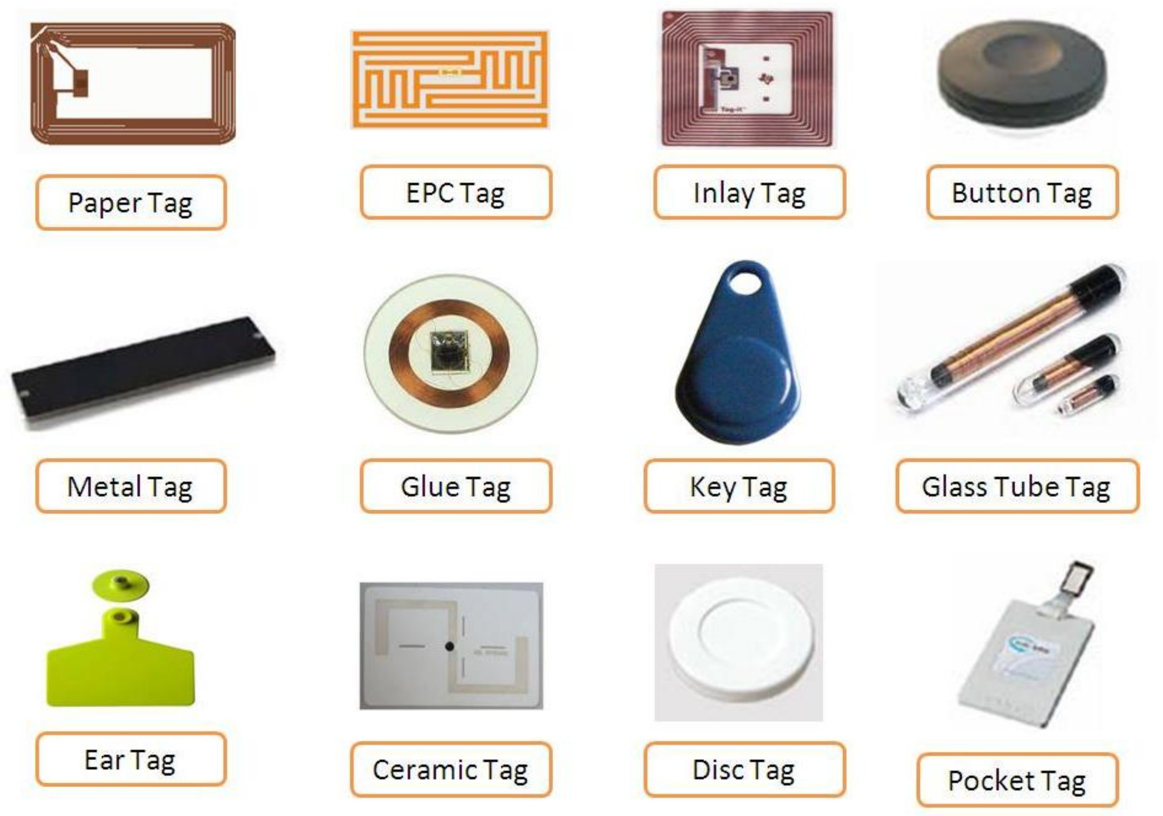


Figure 3.2: RFID Tags

We can divide RFID tags into two categories based on their signal transmission manner: passive and active RFID tags. For passive tags, they collect energy from the RFID readers interrogating radio waves and after the energy in passive tags reach a certain value, passive tags will broadcast a signal which can be catched by RFID readers. So for passive RFID tags, it is not necessary to add a local power source like battery on it. Unlike passive RFID tags, a local power source is necessary for active tags as active RFID tags send signal utilizing the energy from the local power source, which also lead to active RFID tags has larger range of distance of being detected by corresponding RFID readers. Neither active or passive RFID tags are demanded to be within the line of sight of the reader, which is also one of the biggest advantage of RFID technique when compared to barcode technique.

3.2.2 Readers

RFID reader, sometimes also called interrogator or scanner, sends and receives RFID data to and from RFID tags via antennas which can send and receive radio waves. By changing the antennas numbers, detecting ranges, etc. RFID reader can detect the distance, three dimensional location, etc. of corresponding RFID tags.

3.2.3 Host Computer

When the reader receives radio waves it will encode it and transfer those data to a host computer, which may run specialist RFID software or middleware to filter the data and process the data into useful information.

3.3 Hidden Markov Model

A Hidden Markov Model (HMM) is a modeling technique first developed by L. E. Baumin and coworkers. It deals with a situation which you observe a sequence of emissions, but do not know the sequence of states the model went through to generate the emissions. Analyses of hidden Markov models seek to recover the sequence of states from the observed data.

Like a Markov chain, HMM describe a sequence of states. The next state only depends on current state, not the states that precede it. In a Markov chain, the observer can directly observe the state. Therefore the state transition probabilities are the only parameters.

However, HMM has additional parameters, which are emission probabilities. Therefore, the outputs are depended on the states. In a Hidden Markov model, the state is not directly visible, but the output is visible. Therefore the sequence of tokens generated by an HMM gives some information about the sequence of states. Note that the adjective 'hidden' refers to the state sequence through which the model passes, not to the parameters of the model; the model is still referred to as a 'hidden' Markov model even if these parameters are known exactly.

HMM can be described by 5 elements, including 2 state set and 3 probability matrix.

Table 3.1: Components of hidden Markov model

Elements	Meaning
S	Hidden States,which cannot be observed directly
O	Observations,Can be observed directly
π	The probability matrix of hidden states when t=1
T	Transition Matrix,Describe the transfer probability among the stages
E	Emissions Matrix,Describe the output probability of observations in different states

There are three problems of interest:

1. The Evaluation Problem, which compute the probability of a given sequence of observations.

2. The Decoding Problem, which compute the most probable sequence of states, given a sequence of observations.

3. The Learning Problem, which learn best model, given an observation.

There are 3 canonical problems to solve with HMMs:

1. Given the model parameters, compute the probability of a particular output sequence. This problem is solved by the Forward and Backward algorithms.

2. Given the model parameters, find the most likely sequence of (hidden) states which could have generated a given output sequence. Solved by the Viterbi algorithm and Posterior decoding.

3. Given an output sequence, find the most likely set of state transition (maximum likelihood estimation) and output probabilities. Solved by the Baum-Welch algorithm.

Chapter 4

System Design

In this chapter, we introduce the design of an RFID and HMM-based indoor behavior prediction system, which is the the prediction approach we use in our recommendation system and incorporating three subsystem to implement functions of pre-process, training and prediction. Each subsystem is composed by several modules. In the following sections, we elaborate on the function of each module and this subsystem into the whole system.

For the sake of simplifying the problem, we define shopping carts' paths as the same as users' paths, thus we can use RFID tags' paths to represent users' paths. For differing every selling zone clearly without overlapping with other selling zones (in a real situation, zone detection, measurement and differentiation mistakes may occur due to several reasons such as electromagnetic interference, RFID radars' power input not being stable etc.), we then define that every selling zone in an indoor environment is a room containing several merchandise. When a user purchases merchandise, he or she will enter a room and stay. Here we formulate the problem of indoor behavior prediction that we tackle in this work. Table 4.1 summarizes the notations used in this paper.

Definition 1: (**Raw RFID Data**) a set of RFID readings $record_1$, $record_2$,...,$record_n$, where $record_i$ is in the format ($detectorID$, $objectID$, t), and is ordered by detection time t.

Definition 2: (**Aggregated RFID Data**) a set of data has the same format as Raw RFID Data, and the records generated by the same detector are aggregated by an update frequency of once per second.

Definition 3: (**Path record**) a set of detector IDs d_1, d_2,...,d_n that objects pass by from one room (zone) to another.

Table 4.1: Symbolic notations

Elements	Meaning
d_i	RFID detector
r_i	Rooms (Zones)
R_s	Starting room and destination room
G_i	Goods id
$P(s,t)_i$	Paths from Room s to Room t
Seq	A sequence of HMM observations
$ReaderSeq$	A sequence of readers' ID as HMM observations
$RoomSeq$	A sequence of room' ID as HMM observations (states)
$GoodsSeq$	A sequence of gooods' ID as HMM observations
N_s	The number of HMM states
N_o	The number of HMM observations

Definition 4: (**Goods record**) a set of goods IDs G_1, G_2,...,G_n that objects pass by from one room (zone) to another.

Definition 5: (**SAn**) a set of selling area IDs SA_1, SA_2,...,SA_n that contains goods and objects passe by.

Definition 6: (**Rn**) a set of areas can be detected by different readers, we assign IDs R_1, R_2,...,R_n for those areas that objects pass by.

Figure 4.1 shows the overall structure of our system design. The preprocess subsystem deals with raw RFID data. It includes three modules: raw data collector, valid room analysis module and observation sequence generating module. Raw readings are first fed into and processed by raw data collector module, which then provides aggregated readings for each object at every second to the valid room analysis module. The valid room analysis module filters out all the invalid room records by analyzing the time records. The observation sequence generating module transfers the valid room analysis module's outputs to format sequences that can be used in HMM training and testing. The training subsystem incorporates two module: HMM training and route statistics. Through this system, sufficient quantity of records of a person will be used to train HMM. All paths of the person from zone to zone will be input to statistical analysis. The prediction subsystem employs the HMMs and route statistics from the training system to analyze people's indoor behavior. The people's target room and the most probable route will be predicted in that prediction system. The goods prediction subsystem takes space data from space analysis subsystem and goods data and use those data to train HMMs. After training HMMs, generated transform and emission matrixes will be calculated with current data (maximum likelihood estimation), as a result the next purchased goods will be generated. Here we propose three methods to do HMMs training in goods prediction subsystem: 1. Given an output sequence of purchased goods, find the most likely set of (hidden) state transition (maximum likelihood estimation) and output probabilities of next purchased goods (Baum-Welch algorithm). 2. Given an output sequence of paths, find the most likely set of (hidden) state transition (maximum likelihood estimation) and output probabilities of next location (Baum-Welch algorithm). With the output sequence of paths and goods-path relationship information, next purchased goods can be predicted. 3. Given the model parameters purchased goods data as emission sequence and paths data as states sequence, find the most likely sequence of (hidden) paths (states) which could have generated next purchased goods (a given output sequence). This method uses Viterbi algorithm and Posterior decoding.

4.1 Event-Driven Data Collector

In our setting, objects (customer, goods) are moving in an indoor space, such as shown in Figure 4.2. Every object is represented by an RFID tag. Objects' movements are tracked by RFID readers. When the target object enters an RFID reader's detection range, a set of RFID readings will be generated.

This section describes the event-driven data collector which is the front end of the prediction system. The data collector is responsible for storing RFID raw readings in an efficient way for the following training and prediction tasks. A fact we have to know is that when an object passes the detecting range of an RFID reader, there would be a set of raw data generated in the database because of the high reading rate. Considering the characteristic of the Hidden Markov model, we do not need such a high observation frequency. The data

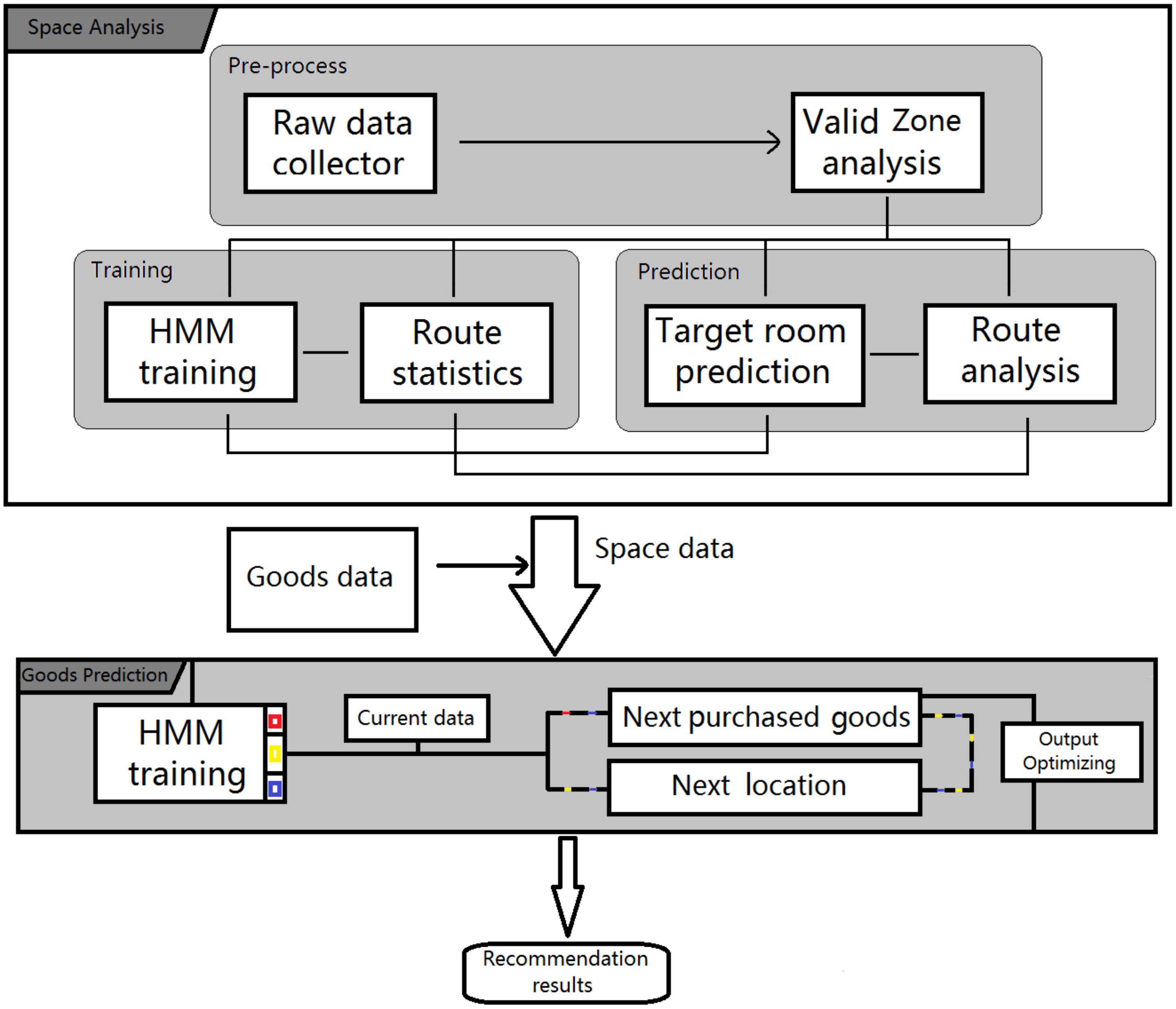


Figure 4.1: Overall Prediction System Structure

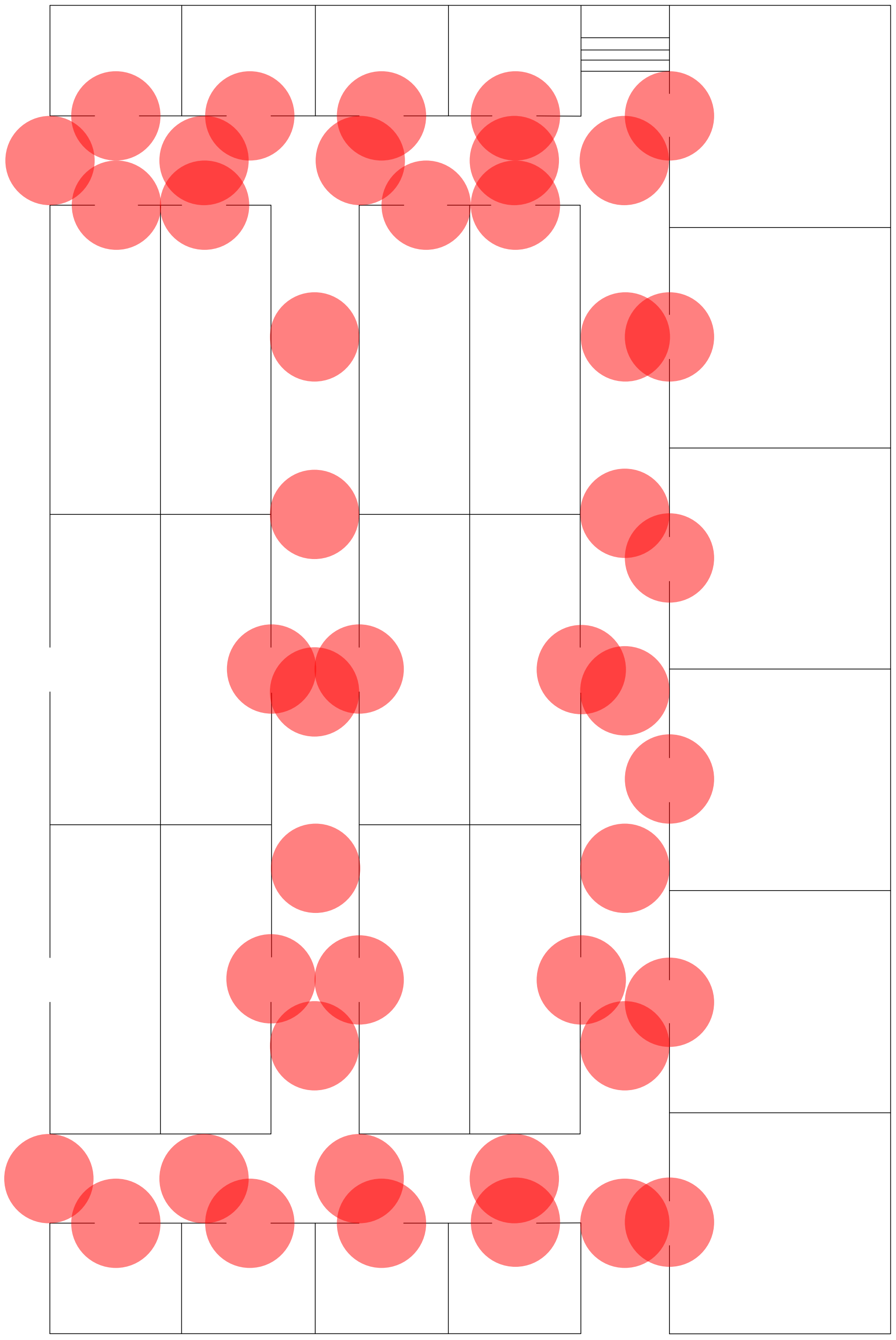


Figure 4.2: Indoor Space

collector module is responsible for aggregating the raw readings to more concise entries with a time unit of one second. The RFID frequency of once per second would be good enough for event observation. Furthermore, aggregating can significantly mitigate the effects of missing readings. Most of the RFID readers' reading rate is near tens of samples per second, and once an object is detected during a second, an entry marking that event is inserted into the aggregated results. It is unlikely that the reader would fail to detect the object during one second, so aggregation can greatly reduce the detecting errors of false negatives.

We define events in this context when the object passes (enter and leave) the reading range of an RFID reader. For example, when an object enters the detection range of D1, the collector will receive a series of raw data signals. The module will not generate a record to the database until the object leave the reading range (the reader stops receiving raw data from D1). The RFID records for each object will be kept in the database with the format ($detectorID$, $objectID$, t).

4.2 RFID based HMM Training and Prediction

In this section, a simple method of RFID based movement prediction and training is provided as the baseline of our improved methods.

A person's indoor movements follow a limited number of regular patterns which map the "states" in the Hidden Markov Model, and in different patterns, the person's moving action follows different schedules that are embodied in the probabilities of target locations. To apply HMM in this scenario, a simple idea is to regard RFID detector's IDs as HMM observations, so that we can analyze a person's moving rules.

With the database record of a person's RFID raw data, an HMM can be trained. We train an HMM for each person we focus on, and the HMM will be used in the prediction module. To train a Hidden Markov Model, there are two methods. The first method is using a sequence of observations and the assorted states. However, we do not know the states that the model went through to generate observations. The second method is using a sequence of observations without the assorted states. We take this method because we can accurately know the observations, which are the detectors that the person passes by.

We should firstly generate a sequence of observations, which is a 1-dimension array. We can just reserve the column of detectorID in the Aggregated RFID Data. Another problem is to determine the parameters. The observation number is the number of RFID readers which is easy to come out with. But what is the state number? We can not use a specific number because it not only depends on the numbers of RFID readers, but also depends on the scenario of the person and the floor. An experiment will be provided in the experiment section to evaluate the state number's influence on the result.

When prediction query is launched, we first get the person's records of the day. We aggregate the RFID data and change it into a sequence of detector IDs by the previous method. Then we can use the HMM which is trained beforehand to compute the posterior state probabilities of the sequence. With the help of transition matrix and emission matrix, the most likely reader that the person will go to next can be calculated.

Algorithm 1 Destination Prediction(T, E, $ReaderSeq$, N_s, N_o)

```
1.  $Pr\_last\_states$ = Dec($T$, $E$, $ReaderSeq$)
2.  initialize array $Pr\_next\_states$
3.  for i = 1 to $N_s$ do do
4.      for j = 1 to $N_s$ do do
5.          $Pr\_next\_states_j \leftarrow Pr\_next\_states_j + Pr\_last\_states_i * T_{i,j}$
6.      end for
7.  end for
8.  initialize array $Pr\_next\_readers$
9.  for i = 1 to $N_s$ do do
10.     for j = 1 to $N_o$ do do
11.         $Pr\_next\_readers_j \leftarrow Pr\_next\_readers_j + Pr\_next\_states_i * E_{i,j}$
12.     end for
13. end for
14. [$maxPr$,$reader$] $\leftarrow$ max($Pr\_next\_readers$)
15. return
```

The algorithm of Destination Prediction is shown in Algorithm 1. Step 1 is to calculate the posterior state probabilities of the input reader ID sequence using the method shown in

Chapter 3. The output Pr_last_states is an N_s-by-1 matrix, where M is the number of states. $Pr\_last\_states_i$ is the conditional probability that the model is in state i when it generates the last reader ID of input sequence, given that the sequence is emitted. From Step 2 to Step 7, we compute the matrix Pr_next_states by multiplying matrix Pr_last_states and transition matrix T. $Pr\_next\_states_i$ represents the probability that the model is in state i when it generates the next ID. Step 8 to Step 13 compute the probabilities of the next reader ID by production of matrix Pr_next_states and emission matrix E.

However, the prediction result of this algorithm is the next RFID's location, which may be quite close to the query point. It is of no significance to predict a destination with a rather near distance. We can use this method multiple times to lengthen the distance. We stop the loop when the person reaches a room because we consider the rooms as significant targets. The algorithm is showed in Algorithm 2.

Algorithm 2 Next Room(Pr_last_states,$RoomIDs$,T,E)

```
1.  repeat
2.     initialize array $Pr\_next\_states$
3.     for i = 1 to $N_s$ do do
4.        for j = 1 to $N_s$ do do
5.           $Pr\_next\_states_j \leftarrow Pr\_next\_states_j + Pr\_last\_states_i * T_{i,j}$
6.        end for
7.     end for
8.     $Pr\_last\_states \leftarrow Pr\_next\_states$
9.     initialize array $Pr\_next\_readers$
10.    for i = 1 to $N_s$ do do
11.       for j = 1 to $N_o$ do do
12.          $Pr\_next\_readers_j \leftarrow Pr\_next\_readers_j + Pr\_next\_states_i * E_{i,j}$
13.       end for
14.    end for
15.    $[maxPr, reader] \leftarrow$ max($Pr\_next\_readers$)
16. until $reader \in RoomIDs$
17. return
```

4.3 Valid Room Analysis module

The previous sections propose an HMM based prediction method. However, this method may cause long-time training because there may be a considerable number of RFID readers. Another shortcoming of this method is the prediction result is the next RFID's location. And if we try to use Algorithm 2 to lengthen the distance, the result would become inaccurate.

To solve the problem, we consider rooms as HMM's observations, so that we can use HMM to analyze a person's movement. To illuminate the concept of "state" in this method, let's walk through an example of a student's activity in the grocery store. David goes to grocery store "Whole Research" every week. Sometimes he stays up late to finish an assignment due or prepare for a exam which is "study state", and sometimes he goes to a party with friends or watches TV shows which is "entertain state", etc. When he is in the "study state", it is easy to know that he would likely to go to the coffee area (in this system, coffee room). And when he is in the "entertain state", he would probably go to his beer zone (room) and snack zone (room). In this scenario, David's movement abides by an HMM which has at least two states: "study state" and "entertain state". In "study state", the coffee room have higher probabilities; and in "entertain state", the beer zone (room) and snack zone (room) have higher probabilities. The HMM is showed in Figure 4.3.

Actually, the scenarios are far more complicated than this example, but the mechanism is similar to it. An HMM with multiple states can describe a person's movement well.

A new problem we face is to find out the valid rooms which a person's really enters. We employ Valid Room Analysis Module to solve it. We will use Figure 4.4 as an example. A premise we want to emphasize here is that all the valid rooms have an RFID reader above the doors.

In Figure 4.4, a person first stays in Room 1, and then he leaves the room and goes to Room 3. The person may be detected by RFID readers D_1, D_2, D_3. After the pre-process, we know that the person has activities in Room 1 and Room 3, but not in Room 2. So we should take R_1 and R_3 as the observations of the HMM. We use a number of this

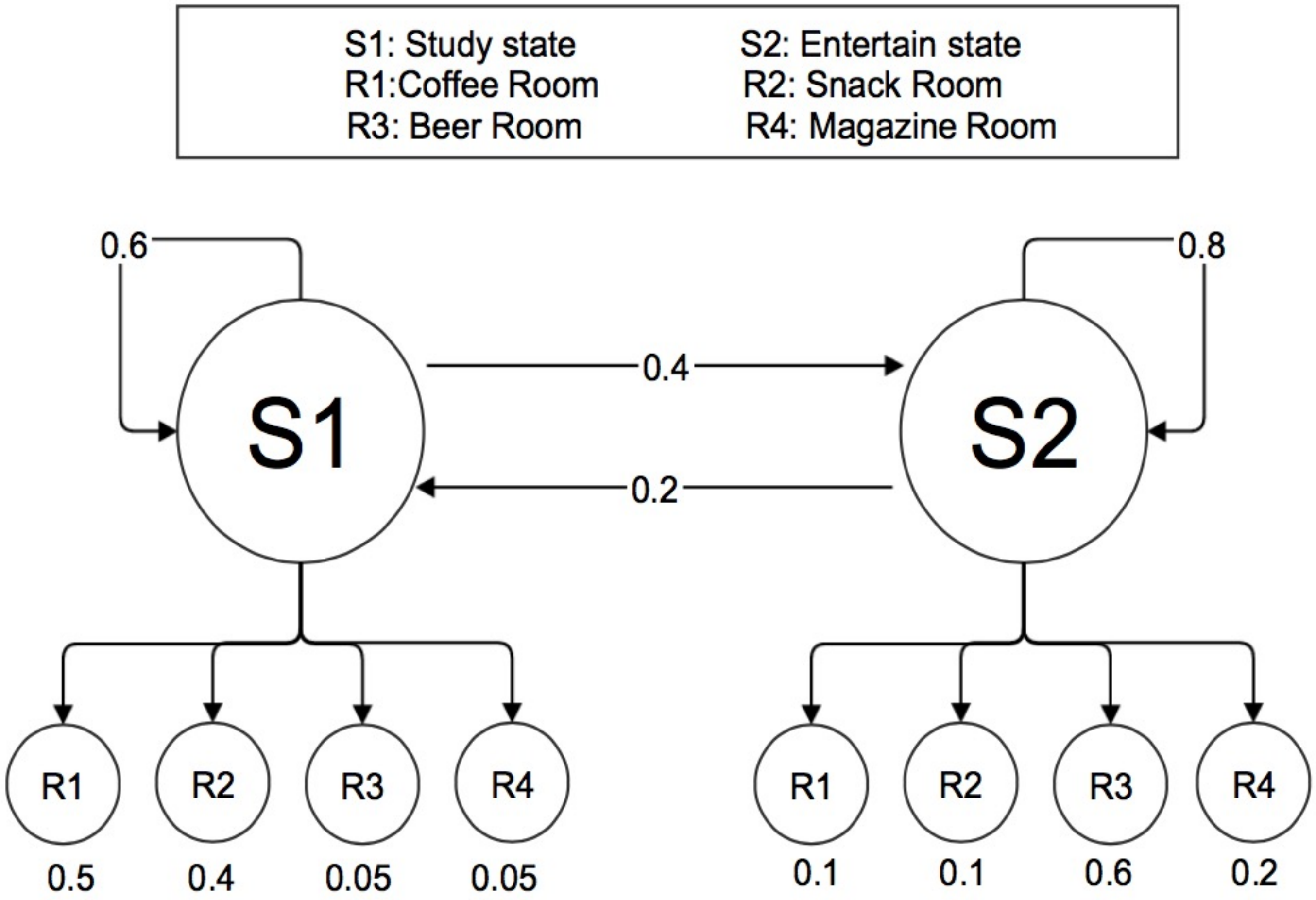


Figure 4.3: Example for HMM

observation sequences to train an HMM. With this model, we can effectively predict the person's movement if we have its previous observation sequence.

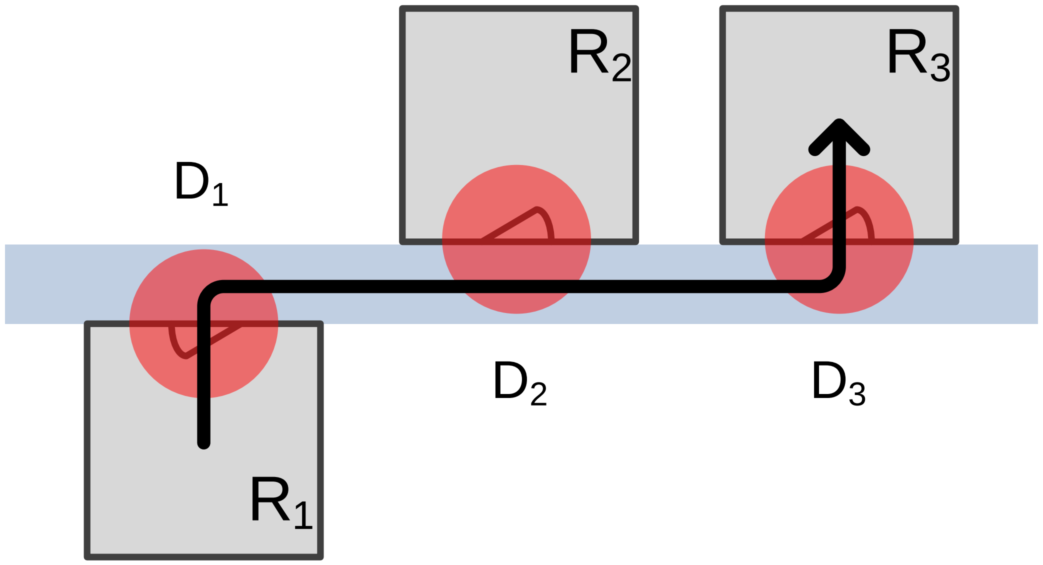


Figure 4.4: Valid Room Analysis

We use following method to estimate whether a person has entered a room. After an initial processing, the RFID reader's sequence may be like this:

...
$<D_1$, object1, $t_1>$
$<D_2$, object1, $t_2>$
$<D_2$, object1, $t_3>$
$<D_2$, object1, $t_4>$
$<D_3$, object1, $t_5>$
$<D_3$, object1, $t_6>$
...

Table 4.2: An example for RFID reader's sequence

The first time the person is detected by D_2 is t_2, and the last time is t_4. If t_4-t_2 is smaller than a threshold (1 min), we will believe that the person does not enter Room 2. Otherwise, we will generate a record $<R_2, object1, t_4>$.

4.4 Target Room Prediction

When we get enough sequence, we can use it to train an HMM and use it for prediction. The method is similar to Algorithm 1. A different place is that Room's ID replaces the Detector's ID in the result.

Algorithm 3 Target Room Prediction(T, E, $RoomSeq$, N_s, N_o)

```
1. $Pr\_last\_states$ = Dec($T$, $E$, $RoomSeq$)
2. initialize array $Pr\_next\_states$
3. **for** i = 1 to $N_s$ do **do**
4.   **for** j = 1 to $N_s$ do **do**
5.     $Pr\_next\_states_j \leftarrow Pr\_next\_states_j + Pr\_last\_states_i * T_{i,j}$
6.   **end for**
7. **end for**
8. initialize array $Pr\_next\_rooms$
9. **for** i = 1 to $N_s$ do **do**
10.   **for** j = 1 to $N_o$ do **do**
11.     $Pr\_next\_rooms_j \leftarrow Pr\_next\_rooms_j + Pr\_next\_states_i * E_{i,j}$
12.   **end for**
13. **end for**
14. [$maxPr$,$room$] $\leftarrow$ max($Pr\_next\_rooms$)
```

First, we calculate the last state probabilities in Step 1. Then we compute the probabilities of the next states from Step 2 to Step 7. $Pr\_next\_states_i$ represent the probability that the model is in state i when it reach next room. Step 8 to Step 13 compute the probabilities of next room by production of matrix Pr_next_states and emission matrix E.

4.5 Path Statistics Module

By the previous method, we can predict people movement through target rooms. However, we only use the filtered RFID readings which contains rooms' locations, the RFID data is not fully used.

In an indoor space, while a person moves from one room to another, the route is likely to be fixed, so the information of previous movement path is helpful for movement prediction. To describe the routes, we can simply sample the coordinates of the moving object. However, the RFID reader can not determine an object's position accurately. Moreover, the readers cannot cover all the indoor space. Considering object movements are limited by the building structures, the movements can be described by specific methods. Here we propose a Reader Based Route Representation method to described the route.

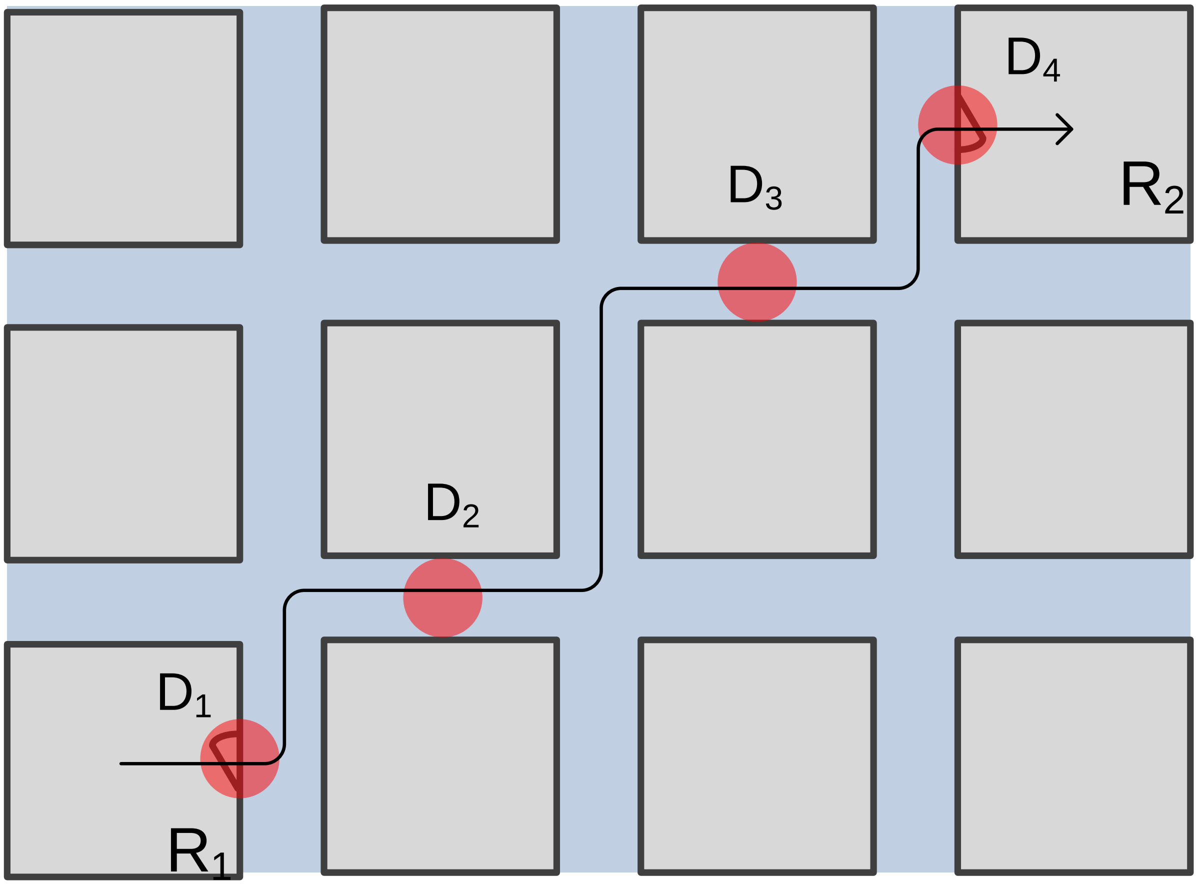


Figure 4.5: Reader Based Route Representation

As shown in Figure 4.5, a person moves from R_1 to R_2 and passes the reader D_1, D_2, D_3 and D_4. We will generate a string "D_1,D_2,D_3,D_4" and search the database for starting room R_1 and ending room R_2. If the database does not have the same route, generate a record (1,2,"1,2,3,4",1) for fields (start:int, end:int, route:string, count:int) in the database. If the database already has this route's record, increase count value by 1.

4.6 Path Analysis Module

In this part, we explain how to improve the prediction accuracy by employing path statistical analysis. By analysing the target room and the person's past movement, we could also find out the path the person will go.

We use the method of statistics. By the pre-processing method above, we can know the rooms in which the person really stays. We use the RFID readers to describe the routes. Once the person passes by an RFID reader, the reader's ID will be recorded, so we can get a sequence of the reader's ID from one room to another. We count these paths by traversing the RFID records of a person, and computing the probabilities of these paths.

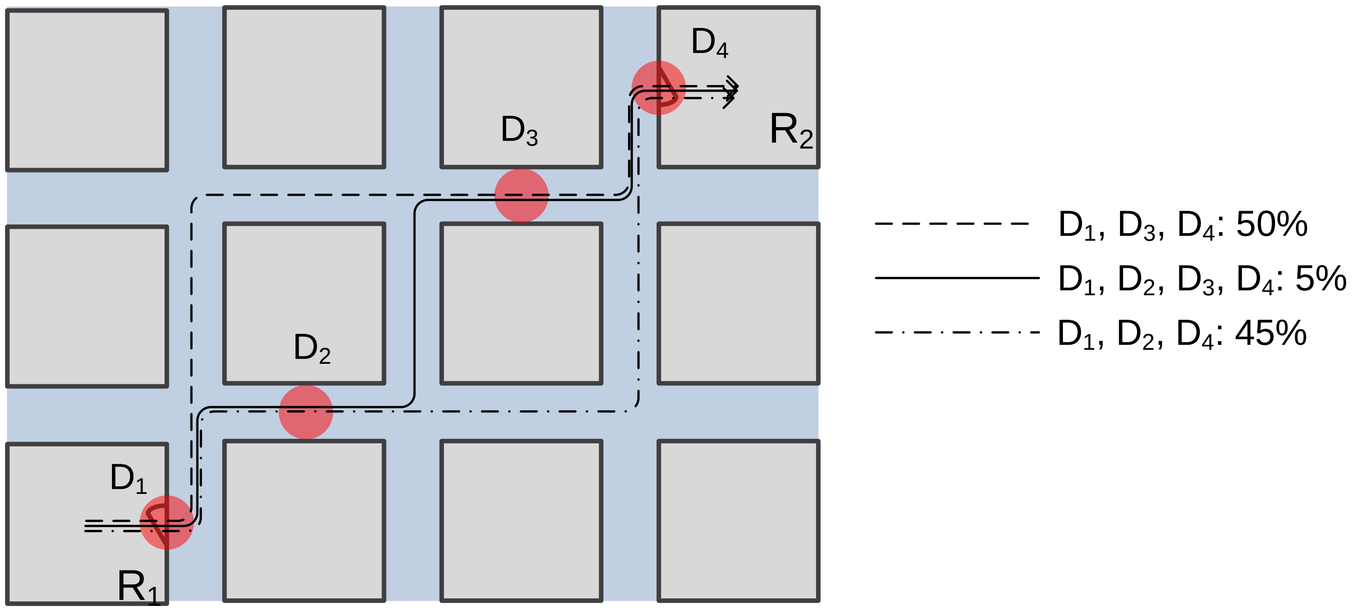


Figure 4.6: Path analyse

For example, in Object 1's statistics, there are three paths from Room R_1 to Room R_2 in Figure 4.6. Path $P(1,2)_1$ passes the readers D_1, D_3 and D_4, with the probability of 50%. Path $P(1,2)_2$ passes the readers D_1, D_2, D_3 and D_4, with the probability of 5%. And path $P(1,2)_3$ passes the readers D_1, D_2 and D_4, with the probability of 45%. Suppose the probability for target Room R_2 calculated by the previous method is p, and the last reader which has detected object 1 is D_1. We can know that all the three paths can be accepted.

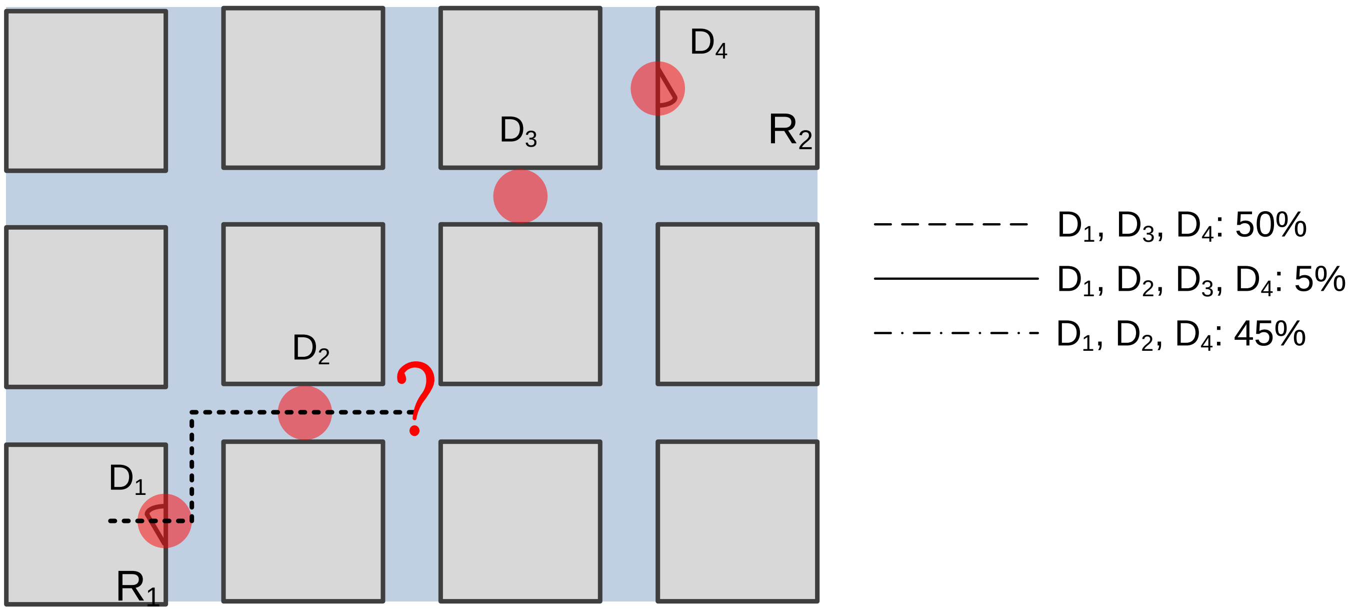


Figure 4.7: Invalid path

However, the last RFID reader which the person passes by is not always D_1. In this case, we can use the RFID sequence to filtrate the paths. For example, as shown in Figure 4.6, we have known the person will move from R_1 to R_2, and after D_1, the person has passed D_2. Because $P(1,2)_1$ will not pass Reader 2, we can deduce that it is impossible for the person to choose $P(1,2)_1$. So we should remove $P(1,2)_1$. The summation of the probabilities for the rest of the paths is 50%, so the fixed probability for target room R_2 is $0.5p$.

Algorithm 4 shows the method of improving the accuracy of room prediction by analyzing the path and path prediction.

4.7 Goods Prediction Module

By previous modules, we can have the space data. Goods prediction module utilize space data and goods data for training HMMs and generating recommendation results. We

Algorithm 4 Path Analyse(R_s, Pr_next_rooms, P, N_r, $ReaderSeq$)

```
1.  $m \leftarrow$ length($ReaderSeq$)
2.  initialize array $Pr\_target\_rooms$
3.  for i = 1 to $N_r$ do do
4.    if i $\neq R_s$ then
5.      for j = 1 to size($P(R_s, i)$) do do
6.        if length($P(R_s, i)_j$) $\geq$ m then
7.          flag $\leftarrow$ 1
8.          for k = 1 to m do do
9.            if $P(R_s, i)_j.d_k \neq ReaderSeq_k$ then
10.             flag $\leftarrow$ 0
11.           end if
12.         end for
13.         if flag = 1 then
14.           $Pr\_target\_rooms_i \leftarrow Pr\_target\_rooms_i + P(R_s, i)_j.pr * Pr\_next\_rooms_i$
15.         end if
16.       end if
17.     end for
18.   end if
19. end for
20. $Pr\_sum \leftarrow$ sum($Pr\_target\_rooms$)
21. if $Pr\_sum > 0$ then
22.   for i = 1 to $N_s$ do do
23.     $Pr\_target\_rooms_i \leftarrow Pr\_target\_rooms_i$ / $Pr\_sum$
24.   end for
25.   [$maxPr$,$nextroom$] $\leftarrow$ max($Pr\_target\_rooms$)
26. else
27.   [$maxPr$,$nextroom$] $\leftarrow$ max($Pr\_next\_rooms$)
28. end if
```

employ the traditional way (the most commonly used way based on statistical probabilities) of recommendation in a grocery store (supermarkets) and algorithms we proposed in this section.

4.7.1 Traditional Recommendation Method

If you go to a grocery store, most likely the only recommend service you can get is from a shop assistant, an advertising screen or a post. These recommendation are based on goods-sell-rates and goods sales events. Advertisements on advertising screen and posters are post to every customer without any differential, shop assistants might be better: they at least do not recommend things you have already brought. These two Bruce force methods are provided in Algorithm 5 and Algorithm 6:

SP (Statistical Probabilities) is the method that use statistical probabilities in historical data to predict the next purchased goods is the one with the highest purchase rate. Advertisements on advertising screen and posts are post to every customer without any differential. This is the most Bruce force but also is the most common method grocery store and supermarkets are using.

Algorithm 5 Statistical Probabilities($historical - Item - List$)

```
1. Sort($PurchasedItemList$, $'descending'$, 2)
2. $predictItem \leftarrow historical - Item - List(0)$ {saves all items in descending order according
   to their historical purchased rate}
```

In Algorithm 5 $historical - Item - List$ is a two dimensional array, the first column of it is goods id, and the second column is purchasing rate of each goods. Sort function here is for sorting the $historical - Item - List$ in descending order according to goods purchasing rates. Then every time the recommendation requests comes, the first goods id in $historical - Item - List$ will be provided as the recommendation result.

SPO (Statistical Probabilities Optimized) is the method that use statistical probabilities in historical data to predict the next purchased goods and the highest purchase rate in the store which has not been added to the cart of this customer. Compared with Algorithm 5, Algorithm 6 takes the goods in customer's carts into consideration, which is more scientific. Because it avoids to repeat recommending the same goods to customers who have already put the goods into their shopping carts.

Algorithm 6 Statistical Probabilities Optimized($Sorted - historical - Item - List$, $PurchasedItemList$)

```
1. $i \leftarrow 0$
2. while $i$ ¡ $Sorted - historical - Item - List.length()$ do
3.   $predictItem \leftarrow Sorted - historical - Item - List(i)$
4.   if $PurchasedItemList$.exist($predictItem$) then
5.     $i - -$
6.   else
7.     $break$
8.   end if
9. end while
```

Through utilizing Algorithm 5, we can have a $Sorted - historical - Item - List$ which contains goods ids and goods purchasing rates and has already been sorted in descending order according to goods purchasing rates. When doing recommendation to a customer, **SPO (Statistical Probabilities Optimized)** method will keep recommending the next-highest-purchasing-rate goods when the recommendation result has already in this customer's shopping cart.

4.7.2 Goods as Emission Sequence Recommendation Method

The **SPO (Statistical Probabilities Optimized)** method, theoretically, is more scientific than **SP (Statistical Probabilities)** method which is currently common used. However, there should be some patterns we can find and utilize to increase the predict hit rate therefore improve the recommendation results' accuracy. However, human's shopping behaviours and the relationship between goods and customers are hard to find, especially there is no extra information for customers when they enter the grocery store. Therefore, hidden markov model can be used in our recommendation module to recognize the hidden patterns and provide prediction for our system.

We believe there are patterns for customer purchased goods. For example, as shows in Figure 4.7.2, if a customer named Liang enters a grocery store, the first thing he buys is goods toothbrush, then there should be a higher chance for him to buy goods toothpaste among goods beer, chips and coffee. However, if the first thing Liang buys is beer, then he may also want to buy chips. Therefore we use goods as observation sequence and use this sequence we can train a HMM. Given an Goods sequence, we can use HMM to find the most likely set of state transition (maximum likelihood estimation) and output probabilities via Baum-Welch algorithm. The output probabilities are assigned to goods id, which we can regard the highest possible goods id as the prediction result.

GH (Goods+HMM) is the method that use goods for the only sequence to train HMM (Need to create/adjust guess transition matrix and guess emission matrix). Details are introduced in Algorithm 7.

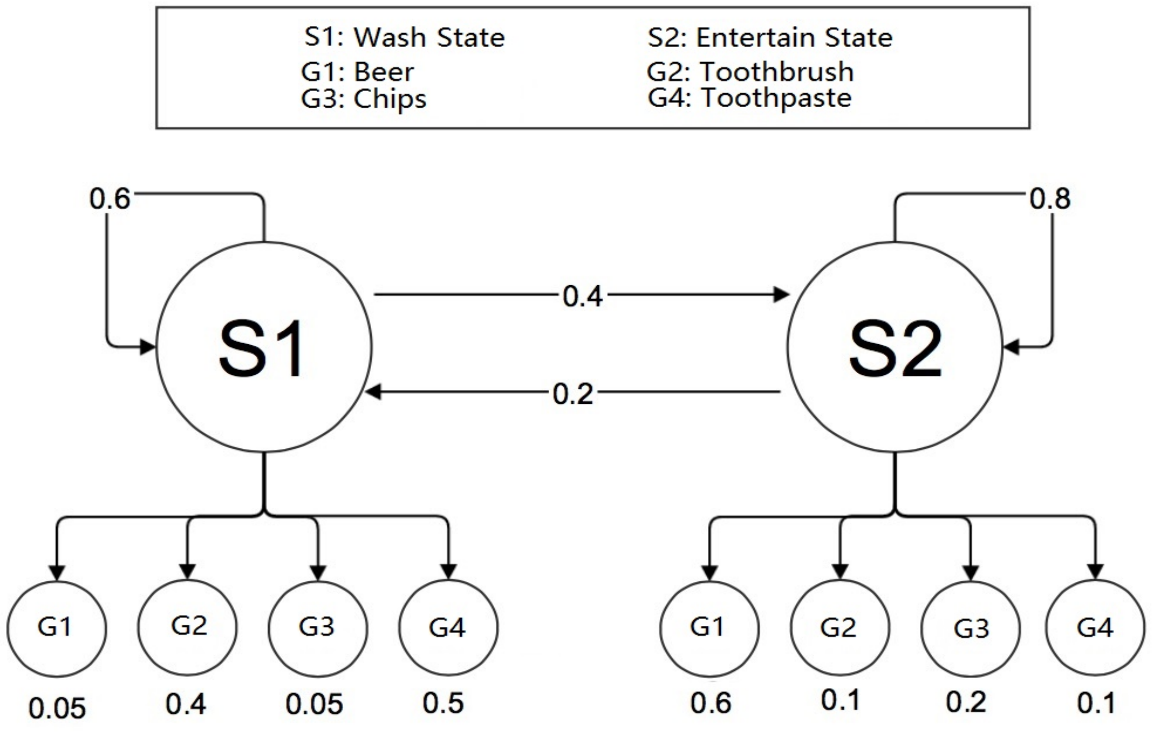


Figure 4.8: Goods + HMM

Algorithm 7 Goods-HMM-Predict($GoodsSequence$, $GoodsPurchased$)

```
1. $EMISGuessMatrix \leftarrow Random(1, numOfSequenceStates)$
2. $TRANSGuessMatrix \leftarrow Random(1, numOfStates)$
3. $Pr\_last\_states$ = Dec($TRANSGuessMatrix$, $EMISGuessMatrix$, $GoodsPurchased$)
4. $N_s \leftarrow$The number of HMM states{Manually set}
5. $N_o \leftarrow$The number of HMM observations {In this case, is goods sequence}
6. initialize array $Pr\_next\_states$
7. for i = 1 to $N_s$ do do
8.     for j = 1 to $N_s$ do do
9.         $Pr\_next\_states_j \leftarrow Pr\_next\_states_j + Pr\_last\_states_i * TRANSGuessMatrix(i, j)$
10.    end for
11. end for
12. initialize array $Pr\_next\_goods$
13. for i = 1 to $N_s$ do do
14.     for j = 1 to $N_o$ do do
15.         $Pr\_next\_goods_j \leftarrow Pr\_next\_goods_j + Pr\_next\_states_i * EMISGuessMatrix(i, j)$
16.     end for
17. end for
18. [$maxPr$, $goods$] $\leftarrow$ max($Pr\_next\_goods$)
```

In Algorithm 7, we firstly initial emission and transition matrixes then we calculate the last state probabilities in Step 3. Then we compute the probabilities of the next states from Step 6 to Step 11. $Pr - next - goods$ represents the probability that the model is in state i when this customer buys next goods. Step 13 to Step 17 compute the probabilities of next purchased goods by production of matrix $Pr - next - states$ and emission matrix $EMISGuessMatrix$.

4.7.3 Path as Emission Sequence Recommendation Method

Similarly, we can regards customers movements as emission sequence to train the HMM. However, the prediction result of this HMM is not goods id, instead, the prediction results here is customer's target location, target location here indicates the next-location a customer will purchase a goods. However, if we can predict a customer's target location, we can then recommend the goods only from that target location for that customer. Space parameter to be consider as an emission sequence is more difficult to understand, but with an example it can be a lot easier. For example, one day Ting is preparing for a dinner, so he goes to vegetable area, and because Ting is not a vegetarian, he also needs some meat, so he then goes to meat area. The other day, Ting is preparing for a party, so he goes to snack area to buy some snacks and then goes to alcohol area to buy some beer. In this case, as in Figure 4.7.3, the areas Ting goes to could be considered as an observation sequence, and using this observation sequence we can train an HMM model to predict the next-area Ting is going to purchase something.

THG (Target Space+HMM+Goods) is the method that use path for the only sequence to train HMM (Need to create/adjust guess transition matrix and guess emission matrix), then predict the target space (room in our model), and then predict and recommend goods in the target space. Details are introduced in Algorithm 8.

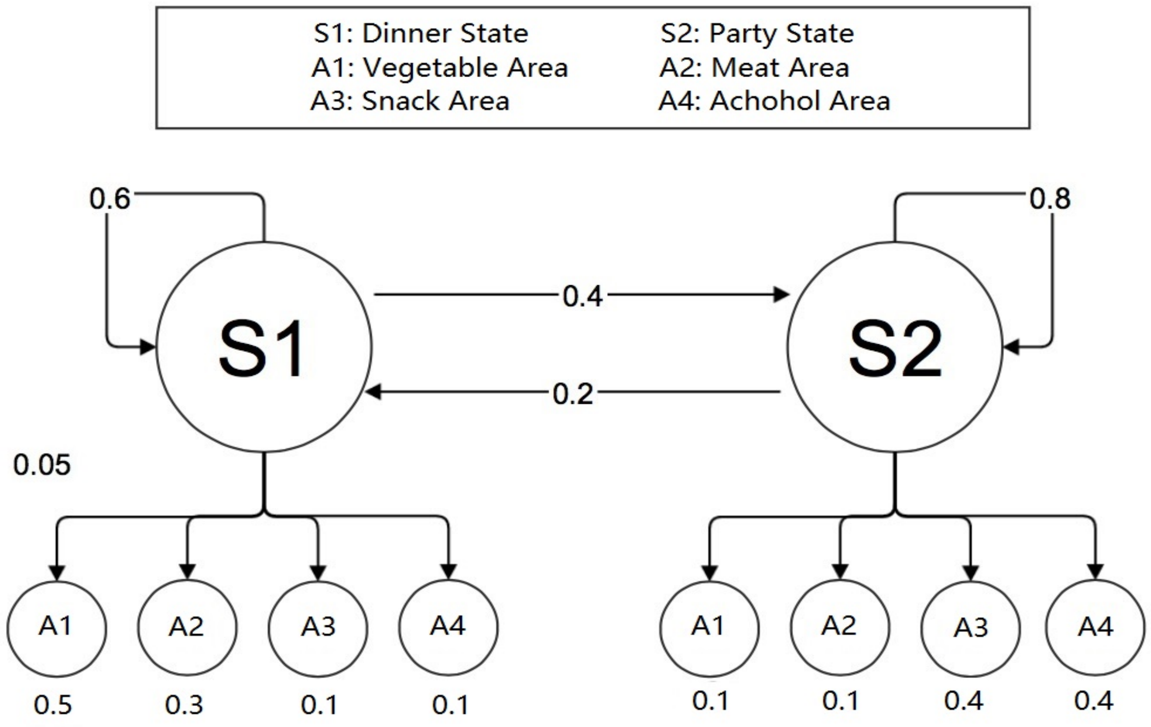


Figure 4.9: Target Space+HMM+Goods

Algorithm 8 TargetSpace-HMM-Goods($PathSequence$,$GoodsInArea$, $GoodsPurchased$)

```
1. $TRANSGuessMatrix \leftarrow Random(1, numOfStates)$
2. $EMISGuessMatrix \leftarrow Random(1, numOfSequenceStates)$
3. $Pr\_last\_states$ = Dec($TRANSGuessMatrix$, $EMISGuessMatrix$, $GoodsPurchased$)
4. $N_s \leftarrow$The number of HMM states{Manually set}
5. $N_o \leftarrow$The number of HMM observations {In this case, is path (Area) sequence}
6. Initialize array $Pr\_next\_states$
7. for i = 1 to $N_s$ do do
8.    for j = 1 to $N_s$ do do
9.       $Pr\_next\_states_j \leftarrow Pr\_next\_states_j + Pr\_last\_states_i * TRANSGuessMatrix(i, j)$
10.   end for
11. end for
12. Initialize array $Pr\_next\_Area$
13. for i = 1 to $N_s$ do do
14.    for j = 1 to $N_o$ do do
15.       $Pr\_next\_Area_j \leftarrow Pr\_next\_Area_j + Pr\_next\_states_i * EMISGuessMatrix(i, j)$
16.    end for
17. end for
18. [$maxPr$, $Area$] $\leftarrow$ max($Pr\_next\_Area$)
19. for Goods in Area do
20.    $PredicatGoods \leftarrow StatisticalProbabilities$(currentAreaProbabilityList,PurchasedItemList)
       More specifically, See Algorithm $StatisticalProbabilities$ 7.
21. end for
```

Similar to Algorithm 7, in Algorithm 8 we firstly initial emission and transition matrixes then we calculate the last state probabilities in Step 3. Then we compute the probabilities of the next states from Step 6 to Step 11. $Pr - next - Area$ represents the probability that the model is in state i when this customer purchased the goods in the next area. Step

13 to Step 17 compute the probabilities of next-purchasing-area by production of matrix $Pr - next - states$ and emission matrix $EMISGuessMatrix$. Different from Algorithm 7, in Algorithm 8 the HMM output is the sequence of next-purchasing-Area ($PrnextArea$), however, we need next-purchased-goods to be our recommendation result, therefore from Step 19 to Step 22 we iterate all the goods in $Pr - next - Area$, then consider the goods with highest sell rate in predicted area as the recommendation result.

4.7.4 Goods Paths as HMM Parameters Recommendation Method

In previous methods, we only consider goods or paths as observation sequence, however, the HMM provides us another algorithm (Viterbi algorithm) to do prediction with both observation sequence and states sequence. In this case, we can use paths as states and goods as observation sequence. For example, as shows in Figure 4.7.4, David goes to a grocery store, when he goes to fruits area to purchase something he has choices to buy apple, orange, watermelon and cherry, and he picks one of them according to his favor which has a very high chance is similar to most people. Similarity, when he goes to snack area he has choices of chips, cookie, candy and Pretzel, and he also picks one of them according to his favor which has a very high chance is similar to most people.

PGH (Path+Goods+HMM) is the method that uses the path as the sate and purchased goods to train HMM, then predicts the next goods this customer is purchasing, and then gives recommendation. Details are introduced in Algorithm 9. The input of this algorithm are current customer's path sequence, the goods in one area, the goods this customer has put into his or her shopping carts, emission matrix and transition matrix produced though using historical purchased goods sequence and customers' walking path sequence training an HMM via Baum-Welch algorithm.

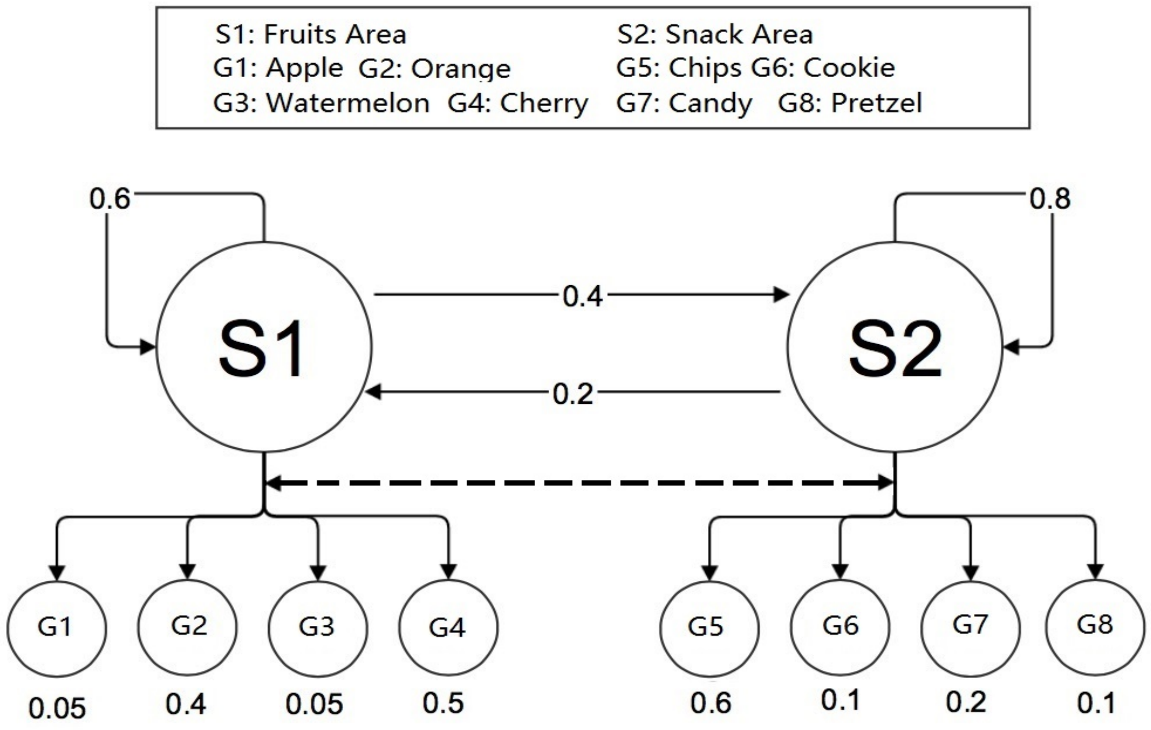


Figure 4.10: Path+Goods+HMM

Algorithm 9 Path-Goods-HMM($PathSequence$, $GoodsInArea$, $GoodsPurchased$, $EMISMatrix$, $TRANSMatrix$)

```
1. $N_s$←The number of HMM states{Manually set}
2. $N_o$←The number of HMM observations {In this case, is path (Area) sequence}
3. Initialize array $Pr\_next\_Area$
4. for i = 1 to $N_s$ do do
5.     for j = 1 to $N_o$ do do
6.         $Pr\_next\_Area_j \leftarrow Pr\_next\_Area_j + Pr\_next\_Area_i * TRANSMatrix(i,j)$
7.     end for
8. end for
9. Initialize array $Pr\_next\_Goods$
10. for i = 1 to $N_s$ do do
11.     for j = 1 to $N_s$ do do
12.         $Pr\_next\_Goods_j \leftarrow Pr\_next\_Goods_j + Pr\_last\_Area_i * EMISMatrix(i,j)$
13.     end for
14. end for
15. [$maxPr$, $Goods$] ← max($Pr\_next\_Goods$)
```

Before Step 1, we use $GoodsSequence$ and $PathStatesSequence$ as parameter to train the HMM (Viterbi algorithm), then we can get $EMISMatrix$ (emission matrix) and $TRANSMatrix$ (transition matrix). Then we compute the probabilities of the next Area from Step 3 to Step 8. $Pr - next - Area$ represents the probability that the model is in state i (area) where this customer will purchase goods. Step 9 to Step 14 compute the probabilities of next-purchasing-goods by production of matrix $Pr - next - Area$ and emission matrix $EMISMatrix$. At last we choose goods id with the highest possibility to be the recommendation result.

Like how we optimized Algorithm 5 to Algorithm 6, we can use the same method to optimize Algorithm 9: ignoring goods already in a customer's shopping cart and recommend next-highest-possibility-goods to this customer. As in Algorithm 10 Step 17, we apply Algorithm 6 here and propose (**PGHO (Path+Goods+HMM+Optimized)**) method.

Algorithm 10 Path-Goods-HMM-Probability-Optimized($PathSequence$,$GoodsInArea$,$GoodsPurchased$
$EMISMatrix$, $TRANSMatrix$)

```
1.  $N_s$←The number of HMM states{Manually set}
2.  $N_o$←The number of HMM observations {In this case, is path (Area) sequence}
3.  Initialize array $Pr\_next\_Goods$
4.  for i = 1 to $N_s$ do do
5.    for j = 1 to $N_s$ do do
6.      $Pr\_next\_Goods_j \leftarrow Pr\_next\_Goods_j + Pr\_last\_Goods_i * EMISMatrix(i,j)$
7.    end for
8.  end for
9.  Initialize array $Pr\_next\_Area$
10. for i = 1 to $N_s$ do do
11.   for j = 1 to $N_o$ do do
12.     $Pr\_next\_Area_j \leftarrow Pr\_next\_Area_j + Pr\_next\_Area_i * TRANSMatrix(i,j)$
13.   end for
14. end for
15. for Goods in Area do
16.   $PredicatGoods \leftarrow StatisticalProbabilities$(AllProbabilityList,PurchasedItemList)
      More specifically, See Algorithm $StatisticalProbabilities$ 7
17. end for{Optimizing through Removing duplexes in purchasing list (recommendation
    goods) based on goods purchasing rates and statical probabilities}
```

Moreover, as recommendation system can provide several recommendation results and we can utilize this rule to provide multiple goods as results to customers. Therefore another optimized method is proposed. **PGHK(PGH3, PGH4, PGH5**) details are described in Algorithm 11. K value here refers to how many goods are included in recommendation results.

Algorithm 11 Path-Goods-HMM-N($PathSequence$,$GoodsInArea$,$GoodsPurchased$, N, $EMISMatrix$, $TRANSMatrix$)

```
1. $N_s$←The number of HMM states{Manually set}
2. $N_o$←The number of HMM observations {In this case, is path (Area) sequence}
3. Initialize array $Pr\_next\_Goods$
4. for i = 1 to $N_s$ do do
5.    for j = 1 to $N_s$ do do
6.       $Pr\_next\_Goods_j \leftarrow Pr\_next\_Goods_j + Pr\_last\_Goods_i * EMISMatrix(i, j)$
7.    end for
8. end for
9. Initialize array $Pr\_next\_Area$
10. for i = 1 to $N_s$ do do
11.    for j = 1 to $N_o$ do do
12.       $Pr\_next\_Area_j \leftarrow Pr\_next\_Area_j + Pr\_next\_Area_i * TRANSMatrix(i, j)$
13.    end for
14. end for
15. [$maxPr$, $Goods$] ← findTopKGoods($Pr\_next\_Goods\_List$,$number$) {findTopKGoods()
    will return K goods based on the prediction possibility from high to low in return matrix,
    for example: for PGH3, number = 3; PGH5, number = 5, etc.}
```

Algorithm 11 is similar to Algorithm 9, the only difference is in Step 16, in which top K goods are picked to be the recommendation results.

Chapter 5

Experiments

In this chapter, we describe the implementation and evaluation of the performance of our indoor grocery store recommendation system using the Hidden Markov Model with location data provided by RFID technologies, and compare the results with the traditional methods of recommendation (Algorithm 5, 6). We implemented the data simulator in OpenJDK 7 and the prediction model in Matlab R2014b. All the experiments were conducted on a computer with 64-bit Windows 10 system, equipping a 2.60GHz CPU and 8GB memory. The setting of our experiment validation includes 12 readers and 10 selling ares as it is showing in Figure 5.1.

5.1 Simulator and Data sets

In the simulated grocery store, we set 12 RFID readers detecting different areas and dividing the whole store into 10 selling areas and 12 reader detected areas. All merchandise are divided into different selling areas of the grocery store. When a customer with a tagged cart and the goods in the cart pass the detection range of a reader, RFID tag ids will be recorded in the database. To ensure the customer's movements simulate reality as closely as possible, we implement a data simulator to generate the testing data.

In the data simulator, each customer has a list which contains all the merchandise he or she is going to purchase in this grocery store. Each kind of merchandise only exist in one selling area in our system. So this customer must go to that specific area to get the merchandise in his or her shopping list. During the time period of purchasing and searching, this customer is able to stay in one or multiple areas, but we record the data into database when this customer (shopping cart) changes to another reader's detecting area. The good's number and good's name (id) in the list and the purchasing order the customer will follow are all randomly generated. However we do have set the possibility of different goods showing on the purchasing list and let the number of purchasing goods follows normal distribution to make our simulation model closer to the reality.

Because we cannot attach RFID tags on customers, we then put RFID tags on shopping carts. We regard one shopping cart as one customer. We also put RFID tags on the goods in the grocery store. When an item's moving path is the same as one shopping cart, we then assume the customer is purchasing this item. For cases in which one customer puts one item into the cart, moves, and then removes the item from the card, we will re-calculate the recommendation list every time this customer passes by a reader's detecting area. To make readers have a better understanding of our simulator, the floor plan and reader settings of our simulated grocery store are showed in Figure 5.1.

The whole simulator consists of 7 components, including Reader Setting Module, Area Setting Module, Goods data initial Module, Goods-Area-Link Module, Purchasing-Lists Generator, Customer-Paths Generator and Output Module. Figure 5.2 shows the relationship of different components in the simulation system.

At the beginning, the reader settings, area settings are initialed via Reader Setting Module and Area Setting Module respectively. In our simulator, areas here indicates different zones in the grocery store, after reader setting being initialed, we assign each areas an unique area ID. Area here has been divided into two types: (1) reader area, which is the area can be detected by a reader; (2) selling area, which indicates areas cannot be detected by readers. Selling areas and reader areas are not overlapped by disinhibition and each selling area is linked to a reader area, and vice versa. This link process is finished in Area Setting Module. Initialed goods information is read by goods data initial module, the format of initial goods data is showing in Figure 5.3.

Each kind of goods has a unique goods id and is saved in RFID tag attached on the goods. The purchasing possibility values here are randomly generated, in reality it can be calculated

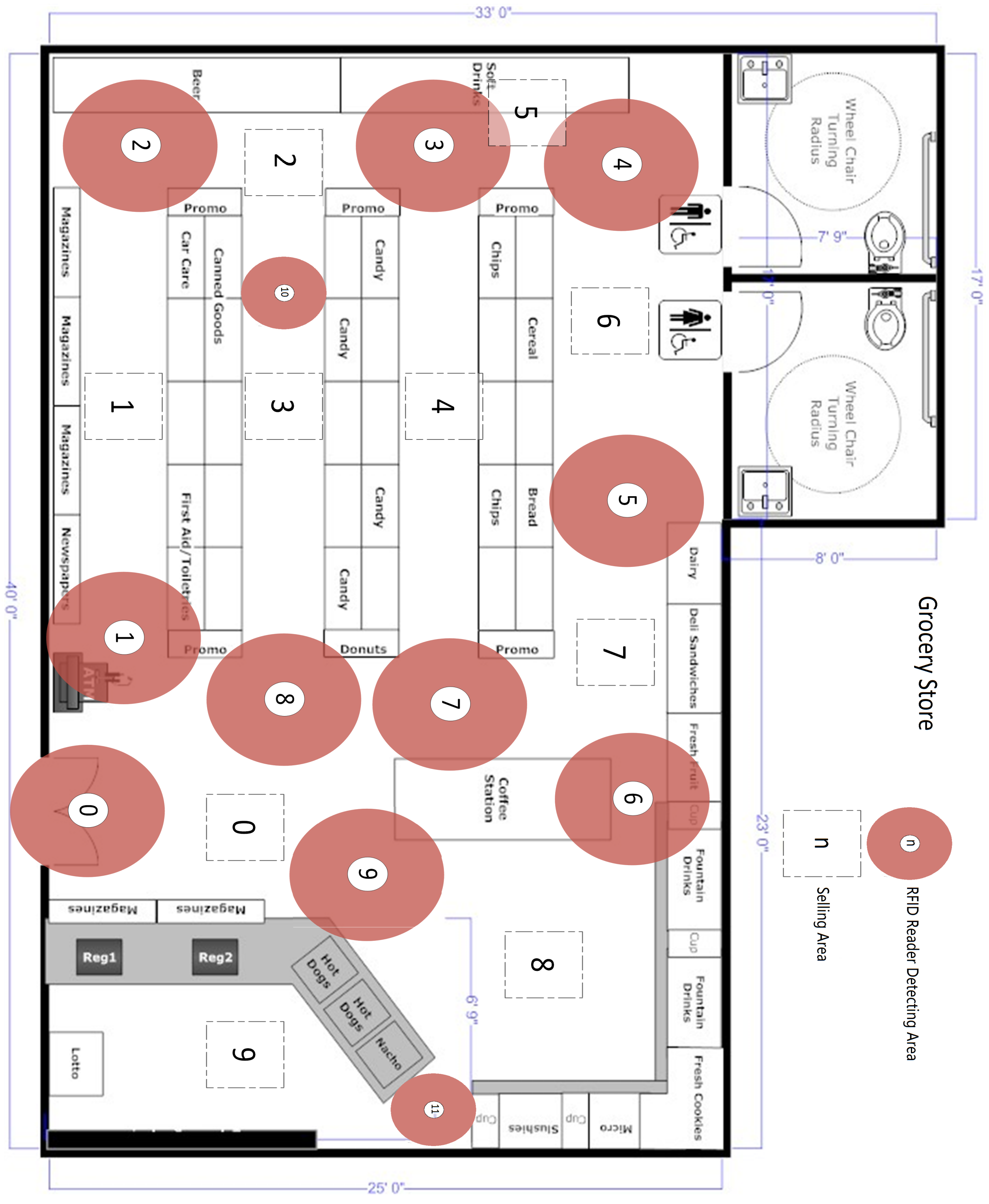


Figure 5.1: Grocery Store Floor Plan and RFID Area Settings

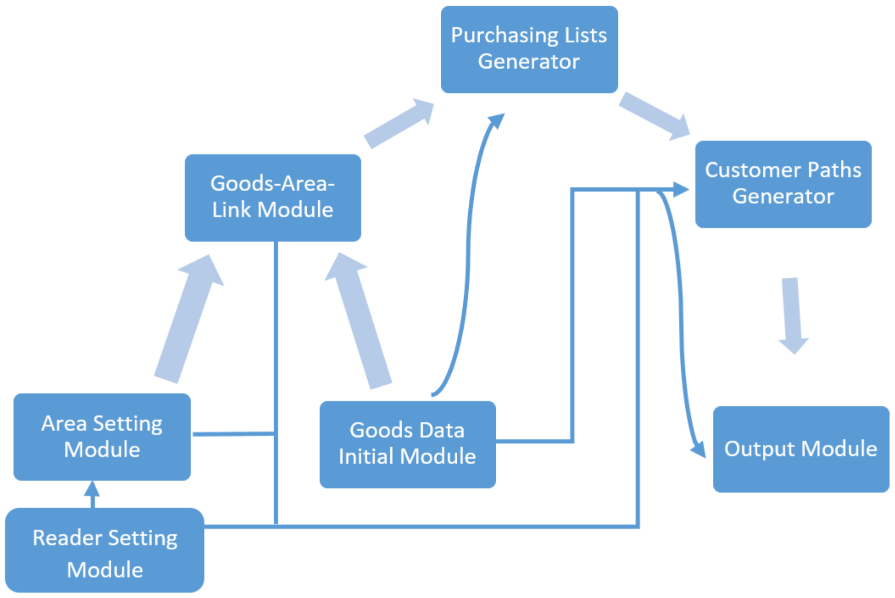
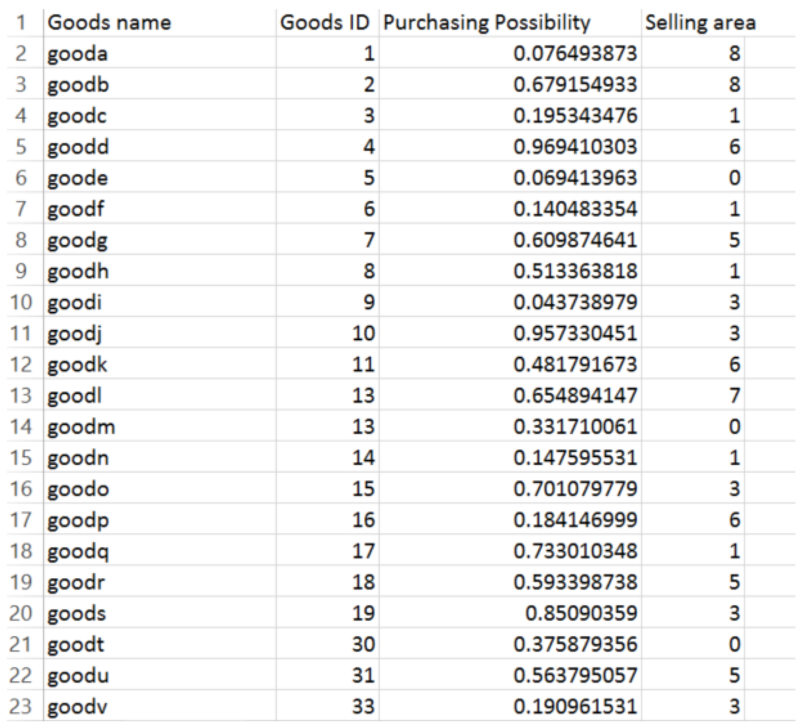


Figure 5.2: The Simulator Structure

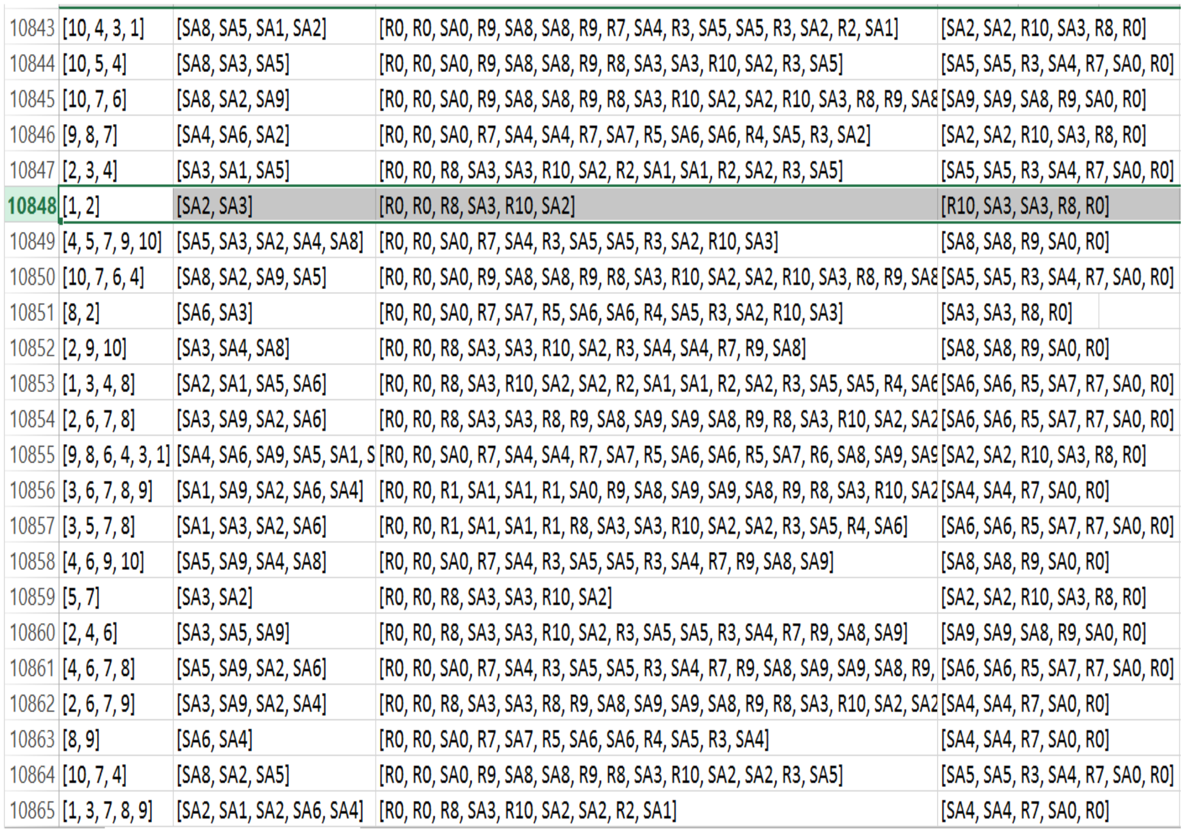


1	Goods name	Goods ID	Purchasing Possibility	Selling area
2	gooda	1	0.076493873	8
3	goodb	2	0.679154933	8
4	goodc	3	0.195343476	1
5	goodd	4	0.969410303	6
6	goode	5	0.069413963	0
7	goodf	6	0.140483354	1
8	goodg	7	0.609874641	5
9	goodh	8	0.513363818	1
10	goodi	9	0.043738979	3
11	goodj	10	0.957330451	3
12	goodk	11	0.481791673	6
13	goodl	13	0.654894147	7
14	goodm	13	0.331710061	0
15	goodn	14	0.147595531	1
16	goodo	15	0.701079779	3
17	goodp	16	0.184146999	6
18	goodq	17	0.733010348	1
19	goodr	18	0.593398738	5
20	goods	19	0.85090359	3
21	goodt	30	0.375879356	0
22	goodu	31	0.563795057	5
23	goodv	33	0.190961531	3

Figure 5.3: Initial Goods Data Format

based on the goods's historical purchasing rate. Each kind of goods is exist and only exist in one selling area in our setting. After we have all the initial goods data, then we can use generate a shopping list for each simulated customer purchasing via list generator. The number of goods follows Normal Distribution. Every "customer" in our system will purchase goods from his or her shopping list, his (her) moving path will generated via customer path generator. The idea of generating a path here is from a customer will first present at R0 (reader area 0), then find the path to the next area contains this customer's next-purchased-goods. Here we use broad first search algorithm in our path-finding part, therefore every time a customer will find the shortest path to the next-purchased goods. After all areas contains goods in a customer's shopping list has been passed, then this customer will return to area R0. However, to make the data closer to the reality, we add some "customers" who do not use the shortest path to find his or her next-purchased-item: we deleting part of goods in some customer's shopping list after his or her paths has been generated, then there are some extra paths for those customers shopping records. The shopping records are generated via output module, sample records are showed in Figure 5.4.

Each record in the output represents a customer's shopping behaviour. The first column is shopping list contains all goods a customer is going to buy. The second column is the target areas contains goods in this customer's shopping list. The third column is the moving path of this customer to purchase all goods in his or her shopping list. The forth column is the path the customer return R0 to pay. Selling area and reader area here represented by SA and R respectively. For example, as shows in line 10848, this customer brought goods id 1,2 and those goods are in selling area 2, 3 respectively; From start point R0, this customer passed R8, then reached SA3 and found goods 1, and then he or she passed R10 and reached SA2 found goods 2. This customer then passed R10, SA3, R8 and returned to R0.



10843	[10, 4, 3, 1]	[SA8, SA5, SA1, SA2]	[R0, R0, SA0, R9, SA8, SA8, R9, R7, SA4, R3, SA5, SA5, R3, SA2, R2, SA1]	[SA2, SA2, R10, SA3, R8, R0]
10844	[10, 5, 4]	[SA8, SA3, SA5]	[R0, R0, SA0, R9, SA8, SA8, R9, R8, SA3, SA3, R10, SA2, R3, SA5]	[SA5, SA5, R3, SA4, R7, SA0, R0]
10845	[10, 7, 6]	[SA8, SA2, SA9]	[R0, R0, SA0, R9, SA8, SA8, R9, R8, SA3, R10, SA2, SA2, R10, SA3, R8, R9, SA8	[SA9, SA9, SA8, R9, SA0, R0]
10846	[9, 8, 7]	[SA4, SA6, SA2]	[R0, R0, SA0, R7, SA4, SA4, R7, SA7, R5, SA6, SA6, R4, SA5, R3, SA2]	[SA2, SA2, R10, SA3, R8, R0]
10847	[2, 3, 4]	[SA3, SA1, SA5]	[R0, R0, R8, SA3, SA3, R10, SA2, R2, SA1, SA1, R2, SA2, R3, SA5]	[SA5, SA5, R3, SA4, R7, SA0, R0]
10848	[1, 2]	[SA2, SA3]	[R0, R0, R8, SA3, R10, SA2]	[R10, SA3, SA3, R8, R0]
10849	[4, 5, 7, 9, 10]	[SA5, SA3, SA2, SA4, SA8]	[R0, R0, SA0, R7, SA4, R3, SA5, SA5, R3, SA2, R10, SA3]	[SA8, SA8, R9, SA0, R0]
10850	[10, 7, 6, 4]	[SA8, SA2, SA9, SA5]	[R0, R0, SA0, R9, SA8, SA8, R9, R8, SA3, R10, SA2, SA2, R10, SA3, R8, R9, SA8	[SA5, SA5, R3, SA4, R7, SA0, R0]
10851	[8, 2]	[SA6, SA3]	[R0, R0, SA0, R7, SA7, R5, SA6, SA6, R4, SA5, R3, SA2, R10, SA3]	[SA3, SA3, R8, R0]
10852	[2, 9, 10]	[SA3, SA4, SA8]	[R0, R0, R8, SA3, SA3, R10, SA2, R3, SA4, SA4, R7, R9, SA8]	[SA8, SA8, R9, SA0, R0]
10853	[1, 3, 4, 8]	[SA2, SA1, SA5, SA6]	[R0, R0, R8, SA3, R10, SA2, SA2, R2, SA1, SA1, R2, SA2, R3, SA5, SA5, R4, SA6	[SA6, SA6, R5, SA7, R7, SA0, R0]
10854	[2, 6, 7, 8]	[SA3, SA9, SA2, SA6]	[R0, R0, R8, SA3, SA3, R8, R9, SA8, SA9, SA9, SA8, R9, R8, SA3, R10, SA2, SA2	[SA6, SA6, R5, SA7, R7, SA0, R0]
10855	[9, 8, 6, 4, 3, 1]	[SA4, SA6, SA9, SA5, SA1, S	[R0, R0, SA0, R7, SA4, SA4, R7, SA7, R5, SA6, SA6, R5, SA7, R6, SA8, SA9, SA9	[SA2, SA2, R10, SA3, R8, R0]
10856	[3, 6, 7, 8, 9]	[SA1, SA9, SA2, SA6, SA4]	[R0, R0, R1, SA1, SA1, R1, SA0, R9, SA8, SA9, SA9, SA8, R9, R8, SA3, R10, SA2	[SA4, SA4, R7, SA0, R0]
10857	[3, 5, 7, 8]	[SA1, SA3, SA2, SA6]	[R0, R0, R1, SA1, SA1, R1, R8, SA3, SA3, R10, SA2, SA2, R3, SA5, R4, SA6]	[SA6, SA6, R5, SA7, R7, SA0, R0]
10858	[4, 6, 9, 10]	[SA5, SA9, SA4, SA8]	[R0, R0, SA0, R7, SA4, R3, SA5, SA5, R3, SA4, R7, R9, SA8, SA9]	[SA8, SA8, R9, SA0, R0]
10859	[5, 7]	[SA3, SA2]	[R0, R0, R8, SA3, SA3, R10, SA2]	[SA2, SA2, R10, SA3, R8, R0]
10860	[2, 4, 6]	[SA3, SA5, SA9]	[R0, R0, R8, SA3, SA3, R10, SA2, R3, SA5, SA5, R3, SA4, R7, R9, SA8, SA9]	[SA9, SA9, SA8, R9, SA0, R0]
10861	[4, 6, 7, 8]	[SA5, SA9, SA2, SA6]	[R0, R0, SA0, R7, SA4, R3, SA5, SA5, R3, SA4, R7, R9, SA8, SA9, SA9, SA8, R9,	[SA6, SA6, R5, SA7, R7, SA0, R0]
10862	[2, 6, 7, 9]	[SA3, SA9, SA2, SA4]	[R0, R0, R8, SA3, SA3, R8, R9, SA8, SA9, SA9, SA8, R9, R8, SA3, R10, SA2, SA2	[SA4, SA4, R7, SA0, R0]
10863	[8, 9]	[SA6, SA4]	[R0, R0, SA0, R7, SA7, R5, SA6, SA6, R4, SA5, R3, SA4]	[SA4, SA4, R7, SA0, R0]
10864	[10, 7, 4]	[SA8, SA2, SA5]	[R0, R0, SA0, R9, SA8, SA8, R9, R8, SA3, R10, SA2, SA2, R3, SA5]	[SA5, SA5, R3, SA4, R7, SA0, R0]
10865	[1, 3, 7, 8, 9]	[SA2, SA1, SA2, SA6, SA4]	[R0, R0, R8, SA3, R10, SA2, SA2, R2, SA1]	[SA4, SA4, R7, SA0, R0]

Figure 5.4: Simulator Sample Output

5.2 Experiment Result and Evaluation

For different initial goods category number (10, 20, 30, 40, 50), 5000 out of 10000 records are applied to train our model and another 5000 records are used for evaluating the prediction and recommendation performance of each algorithm. The results are as follows (we do each experiment 10 times and use the average hit rates for comparison):

We compare each proposed algorithm with traditional methods Algorithm 5, Algorithm 6, as it is showing in Figure 5.2, 5.2, 5.2, 5.2, 5.2, 5.2, 5.2, our proposed methods have higher hit rates comparing with traditional methods, especially when goods category grows, our proposed methods hit rates decrease slower than traditional methods.

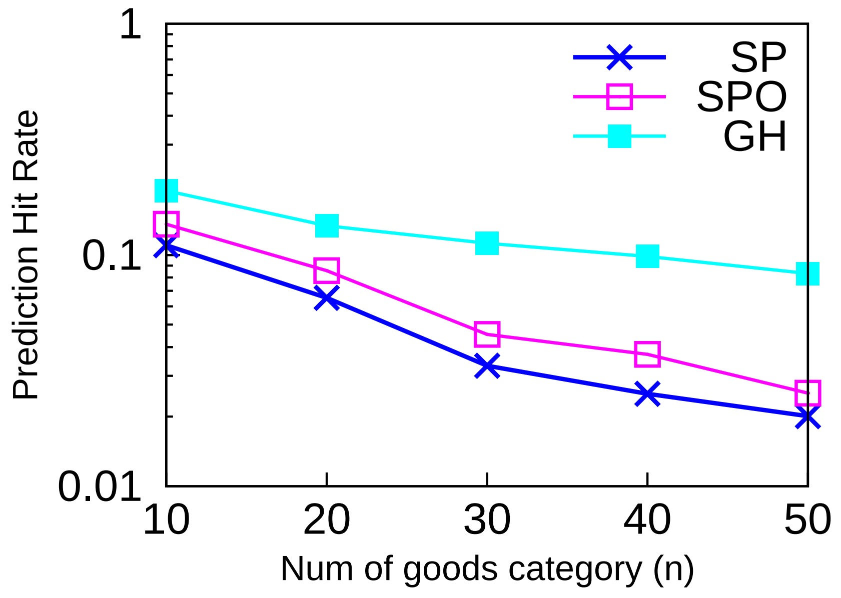


Figure 5.5: GH Hit Rates

Setting the initial goods category number to 28 and comparing different algorithm's hit rates. As Figure 5.2 shows, through adding location as a parameter into our system, we improve the prediction accuracy of the next goods customers going to purchase. However, a recommendation system can have multiple values polled to the user and let the user to choose. Through utilizing this rule, we improve our recommendation system by adding the numbers of goods recommendable to the user. What's more, for different K number in

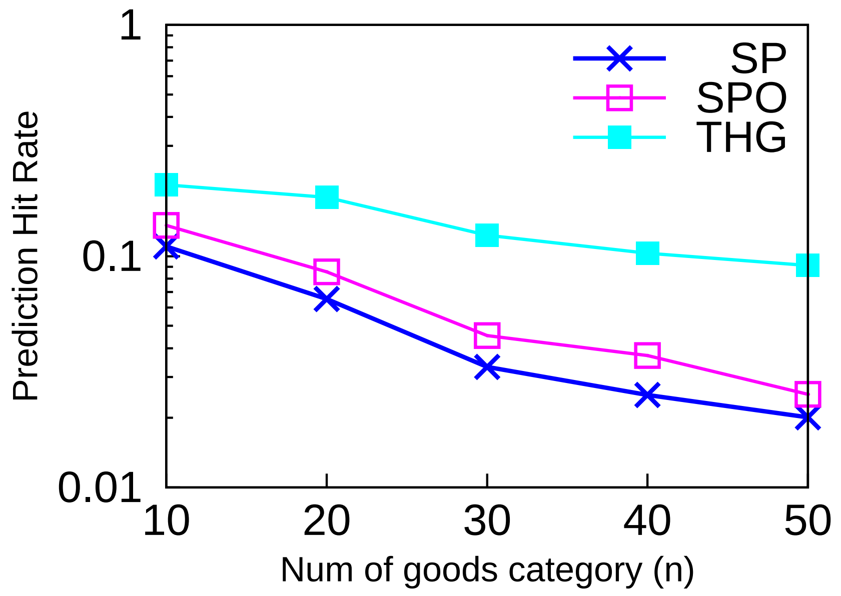


Figure 5.6: THG Hit Rates

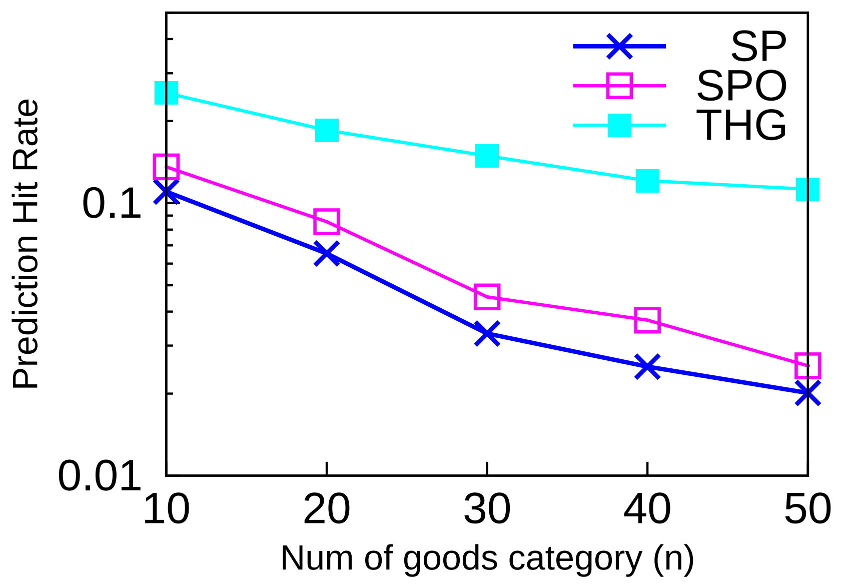


Figure 5.7: PGH Hit Rates

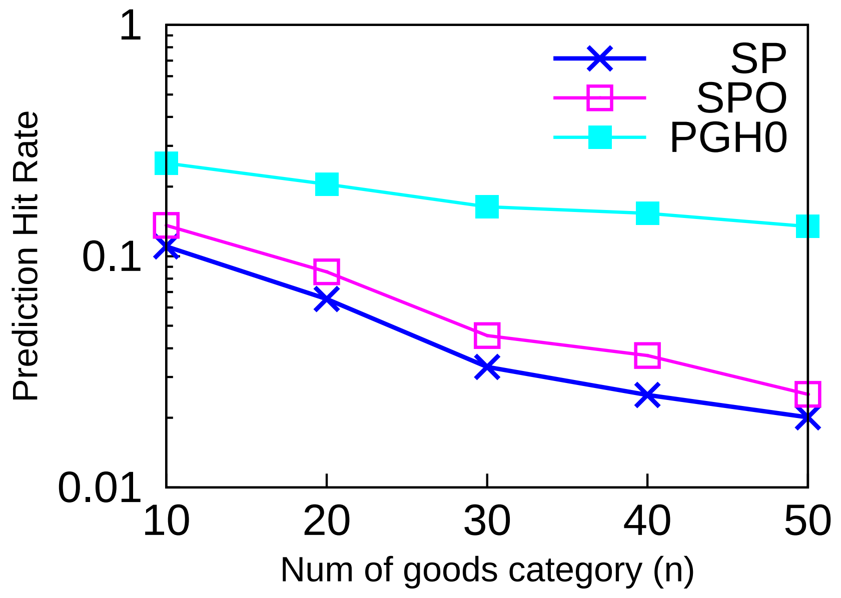


Figure 5.8: PGHO Hit Rates

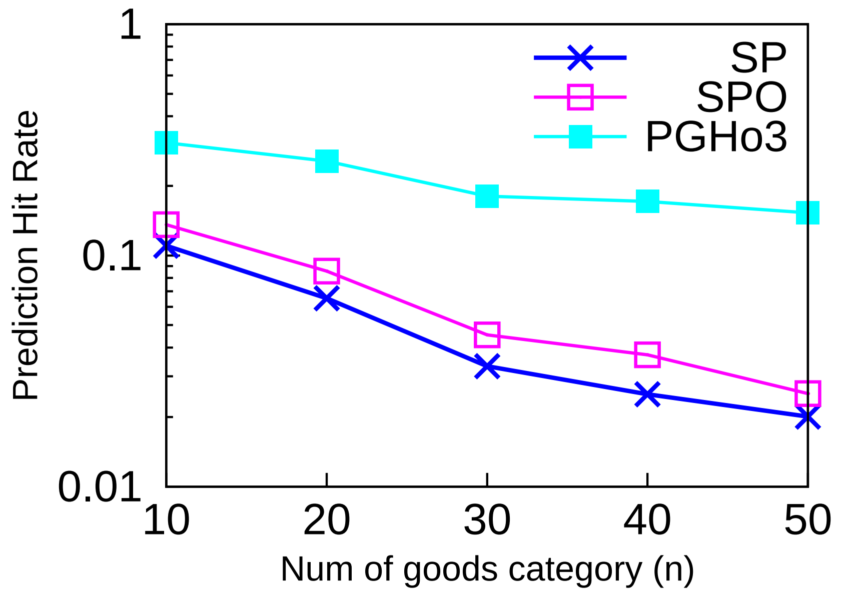


Figure 5.9: PGHO3 Hit Rates

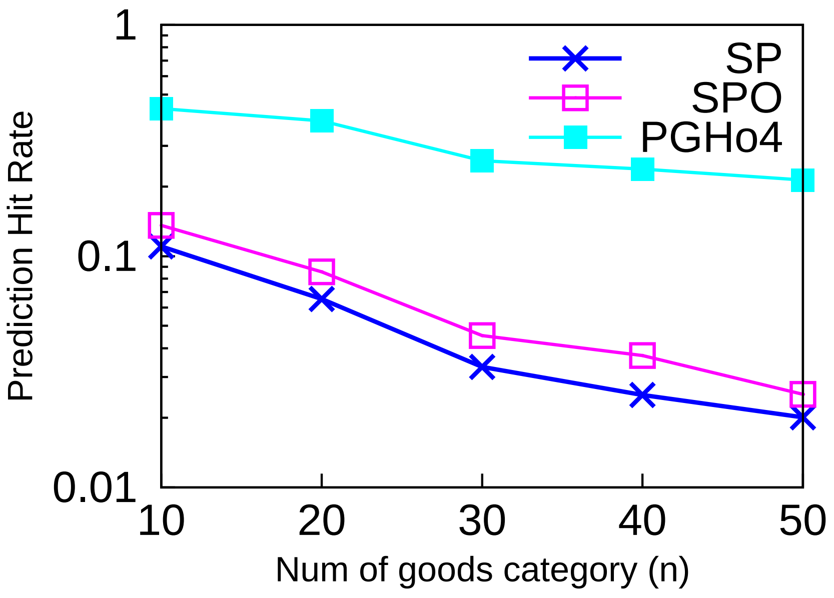


Figure 5.10: PGHO4 Hit Rates

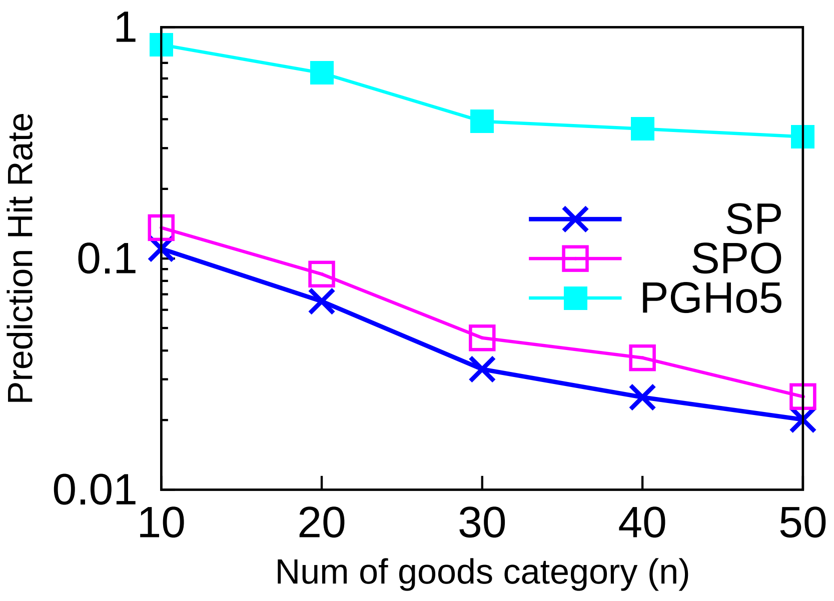


Figure 5.11: PGHO5 Hit Rates

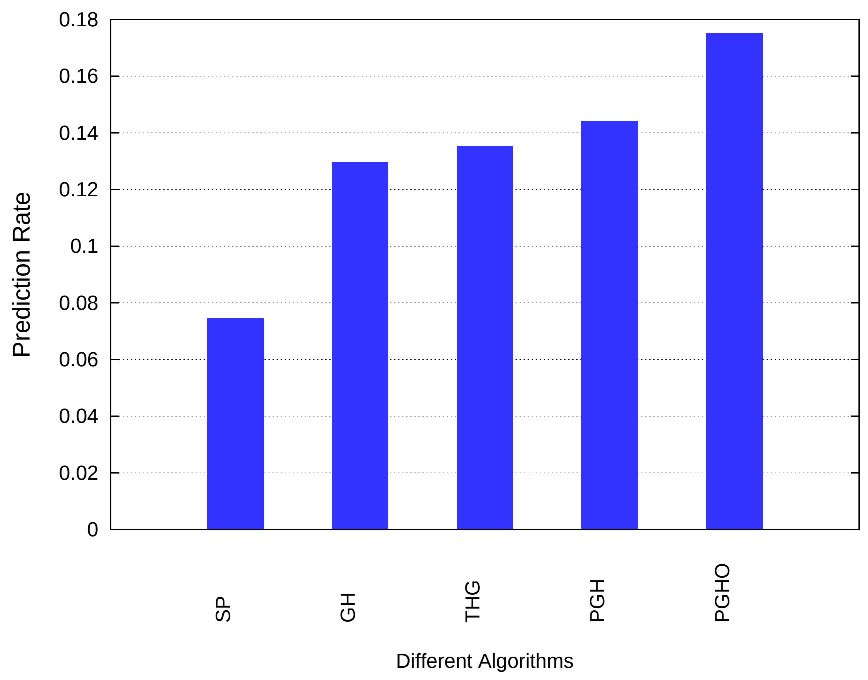


Figure 5.12: Prediction Hit Rates for 28 Kinds of Goods

Algorithm 11, we compare the performance in Figure 5.2. The hit rate of recommendation results considerably grows as the K number grows.

Figure 5.2 shows (setting initial goods category number to 28) total time cost for 100 times of executing different methods to do recommendation. We do not count the HMM training time here because there is no need for training the HMM every time a new recommendation request comes. Therefore as recommendation requests number grows to a very large number, the one-time training time of HMM has very little impact on the total reaction time of the system. Traditional methods use little time than our proposed methods, however, the time of our methods still very small and are acceptable as for trading higher recommendation hit rate. **PGH** and **PGHO** have a lower time cost comparing to **GH** and **THG**.

Adjusting the initial number of states in the HMM is another way to improve recommendation performance. In Figure 5.2, it is observed that the hit rates are elevated when number of initial states increases.

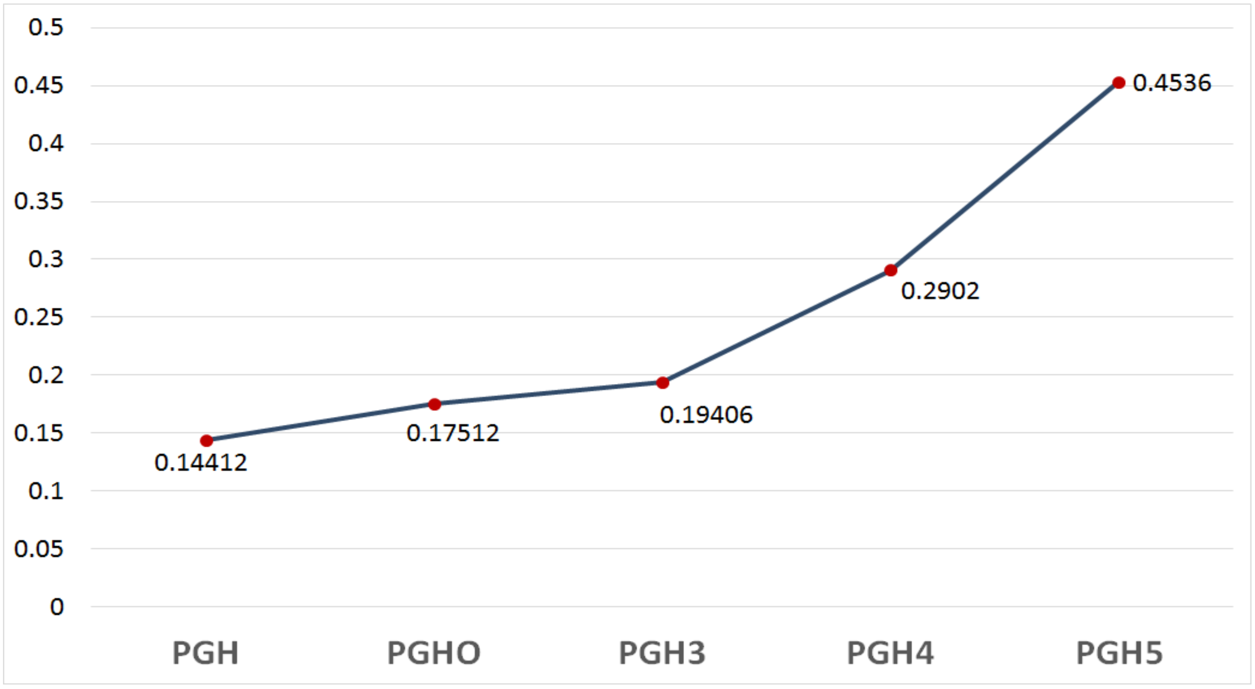


Figure 5.13: Path-Goods-HMM and Optimized Recommendation Methods Hit Rate

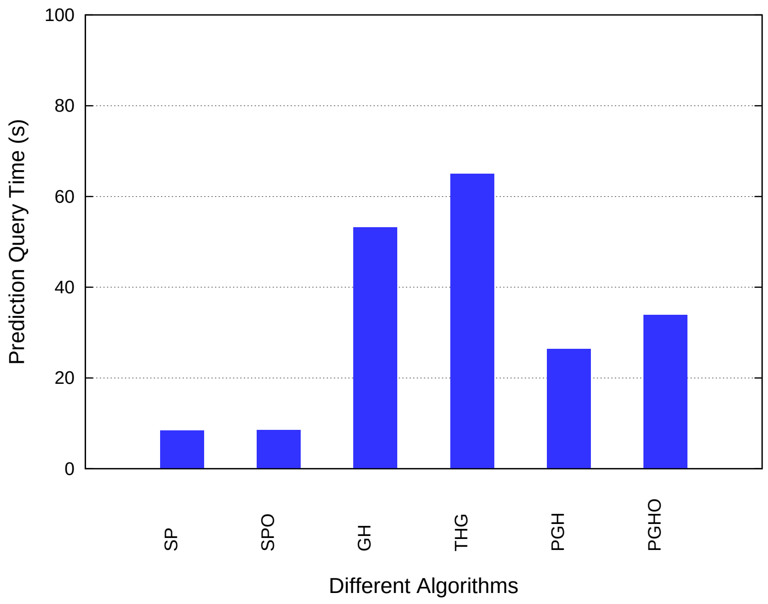


Figure 5.14: Time Efficiency

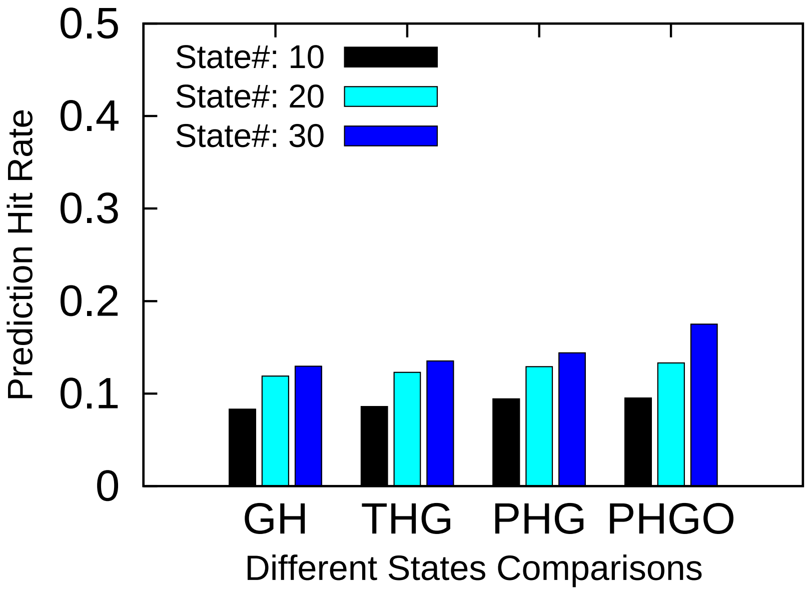


Figure 5.15: HMM states evaluation

Chapter 6

Conclusion and Future Work

On the basis of the result of our simulation and experiment comparisons, it is safe to conclude that our proposed approach, RFIF based PGHN, is an effective solution for implementing a recommendation system regarding offline stores. With the help of RFID technologies, we can track customer's purchasing and moving behaviours by tracking shopping carts and goods which are tagged with RFID tags and then recommend goods for customers based on our historical data and this specific customer's current data including the goods in his or her shopping cart, his or her moving trace and current location. Because this is a recommendation problem instead of a traditional predicting problem, we can provide several goods for the user as the recommendation results. As a result, the hit rate of our system has been improved significantly, which makes this recommendation system more practical to be applied in reality. Furthermore, our system uses a grocery store as an example for simulation and evaluation, however, it is extensible, as we can easily apply our system in various scenarios simply following the same proposed solution.

On the other hand, there are still several aspects we can do in this project in the future. First, although we have successfully simulated our idea via our simulator, there may be differentials between a code implemented simulation and reality; it is really difficult for us to make our simulated data exactly the same as human behaviours. We can still improve the simulator to make it much closer to the reality. Even more ideally, we can find real data in an RFID technology equipped grocery store, have our recommendation system to be implemented, and extensively test our system from a real grocery store, the result can be more convincible. What's more, one of the limitations of the HMM model is that the training of the model is extremely time consuming, so we cannot update the emission matrix or transition matrix quite frequently, which might lead to lower accuracy for the initial phase of using our system. The approach of recommending goods for customers with only few numbers of historical data can be an appealing attribute which we would love to add into our system in the future.

Bibliography

[1] G. Adomavicius and A. Tuzhilin. Toward the next generation of recommender systems: A survey of the state-of-the-art and possible extensions. *Knowledge and Data Engineering, IEEE Transactions on*, 17(6):734–749, 2005.

[2] R. B. Allen. User models: theory, method, and practice. *International Journal of man-machine Studies*, 32(5):511–543, 1990.

[3] C. Archibald, Y. Zhang, and J. Liu. Human motion prediction for indoor mobile relay networks. In *IEEE 13th International Conference on High Performance Computing and Communication-s*, pages 989–994, 2011.

[4] M. Brunato, R. Battiti, A. Villani, and A. Delai. A location-dependent recommender system for the web. 2002.

[5] P. Brusilovski, A. Kobsa, and W. Nejdl. *The adaptive web: methods and strategies of web personalization*, volume 4321. Springer Science & Business Media, 2007.

[6] L. Chen, G. Chen, M. Jin, and E. Wu. A novel rss-based indoor positioning algorithm using mobility prediction. In *39th International Conference on Parallel Processing Workshops*, pages 549–553, 2010.

[7] Coresonant.com. Rfid tags for solar module india, 2014. [Online; accessed 5-April-2016].

[8] A. Felfernig, K. Isak, K. Szabo, and P. Zachar. The vita financial services sales support environment. In *PROCEEDINGS OF THE NATIONAL CONFERENCE ON ARTIFICIAL INTELLIGENCE*, volume 22, page 1692. Menlo Park, CA; Cambridge, MA; London; AAAI Press; MIT Press; 1999, 2007.

[9] L. Ghouti, T. R. Sheltami, and K. S. Alutaibi. Mobility prediction in mobile ad hoc networks using extreme learning machines. *Procedia Computer Science*, 19(0):305–312, 2013.

[10] L. Grossman. Facebook, pandora lead rise of recommendation engines - time. 2010.

[11] L. Grossman. How computers know what we wantbefore we do. *Time Magazine*, 27, 2010.

[12] P. Gupta, A. Goel, J. Lin, A. Sharma, D. Wang, and R. Zadeh. Wtf: The who to follow service at twitter. In *Proceedings of the 22nd international conference on World Wide Web*, pages 505–514. International World Wide Web Conferences Steering Committee, 2013.

[13] J. Kolodziej, S. U. Khan, L. Wang, N. Min-Allah, S. A. Madani, G. N, and H. Li. An application of markov jump process model for activity-based indoor mobility prediction in wireless networks. In *Frontiers of Information Technology*, pages 51–56, 2011.

[14] J. Kolodziej and F. Xhafa. Utilization of markov model and non-parametric belief propagation for activity-based indoor mobility prediction in wireless networks. In *International Conference on Complex, Intelligent and Software Intensive Systems*, pages 513–518, 2011.

[15] G. Linden, B. Smith, and J. York. Amazon. com recommendations: Item-to-item collaborative filtering. *Internet Computing, IEEE*, 7(1):76–80, 2003.

[16] P. Melville and V. Sindhwani. Encyclopedia of machine learning, 2010.

[17] R. J. Mooney and L. Roy. Content-based book recommending using learning for text categorization. In *Proceedings of the fifth ACM conference on Digital libraries*, pages 195–204. ACM, 2000.

[18] M.-H. Park, J.-H. Hong, and S.-B. Cho. Location-based recommendation system using bayesian users preference model in mobile devices. In *Ubiquitous Intelligence and Computing*, pages 1130–1139. Springer, 2007.

[19] F. Ricci, L. Rokach, and B. Shapira. *Introduction to recommender systems handbook*. Springer, 2011.

[20] B. Sarwar, G. Karypis, J. Konstan, and J. Riedl. Application of dimensionality reduction in recommender system-a case study. Technical report, DTIC Document, 2000.

[21] E. Spertus, M. Sahami, and O. Buyukkokten. Evaluating similarity measures: a large-scale study in the orkut social network. In *Proceedings of the eleventh ACM SIGKDD international conference on Knowledge discovery in data mining*, pages 678–684. ACM, 2005.

[22] W. Wade. A grocery cart that holds bread, butter and preferences. *New York Times*, 84:144, 2003.

[23] Wikipedia. Collaborative filteringwikipedia, the free encyclopedia, 2016. [Online; accessed 4-April-2016].

[24] Wikipedia. Radio-frequency identification — wikipedia, the free encyclopedia, 2016. [Online; accessed 5-April-2016].

[25] Wikipedia. Recommender system — wikipedia, the free encyclopedia, 2016. [Online; accessed 4-April-2016].

[26] W.-S. Yang, H.-C. Cheng, and J.-B. Dia. A location-aware recommender system for mobile shopping environments. *Expert Systems with Applications*, 34(1):437–445, 2008.