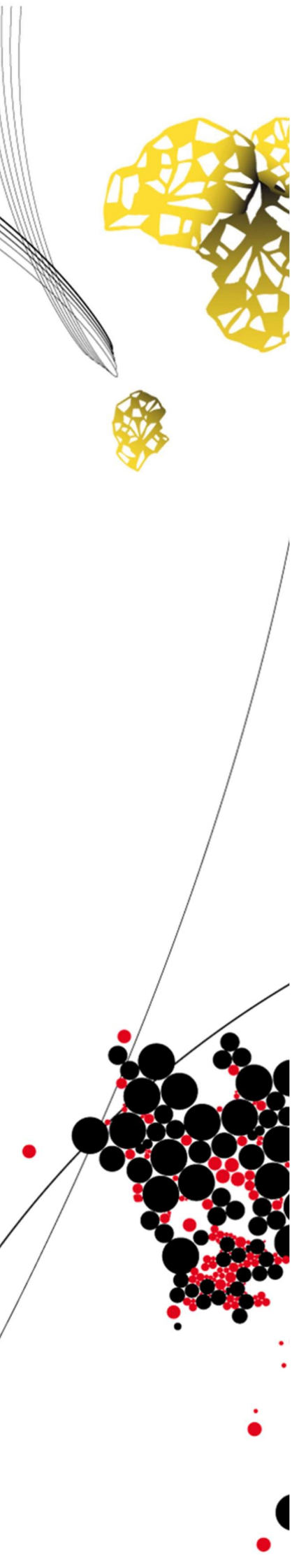

UNIVERSITY OF TWENTE.

Faculty of Electrical Engineering, Mathematics & Computer Science

PHILIPS

Lighting

Inference of social networks in office buildings based on sensor data from lighting systems

Rico F.J. van Lingen
M. Sc. Thesis
November 2017

Graduation committee:
Prof. dr. R.J. Boucherie (UT)
Prof. dr. N. Litvak (UT,TU/e)
Prof. dr. ir. W.L. IJzerman (Philips Lighting, TU/e)
Dr. B. Manthey (UT)

Supervisors:
Prof. dr. N. Litvak (UT,TU/e)
Prof. dr. ir. W.L. IJzerman (Philips Lighting, TU/e)

Stochastic Operations Research
Department of Applied Mathematics
Faculty of Electrical Engineering,
Mathematics and Computer Science
University of Twente

Abstract

The aim of this thesis is to investigate to what extend it is possible to infer a social network based on sensor data from lighting systems in an office building.

The first part of this thesis is devoted to the development of a model with which we can generate sensor data. We have two reasons for generating our own data. The first reason is that there is no data available for this project. However, the main reason for generating our own data is the fact that we want to control the input. This enables us to evaluate the performance of the inference methods we will present. We will give a brief overview of literature on the simulation of occupancy in office buildings. Next, we will present a model with which sensor data will be generated. In our model, the main focus will be on how we can incorporate the input (a representation of the interaction amongst the occupants) properly into the process to generate sensor data. We will make use of a graph in which all possible meetings are represented by vertices. Two meetings will be connected if they share an occupant and hence cannot be scheduled together. By generating a sequence of maximal independent sets of this graph, we can generate a meeting schedule. The mechanism to generate this sequence will make use of the interaction amongst the occupants. Using this meeting schedule, we will generate sensor data.

The second part of this thesis is devoted to the analysis of the sensor data. We will present three examples of an office building with six occupants that will be used to evaluate the performance of the inference methods. Next, we discuss the main challenge for the inference that arises from the data we have generated. We will propose a series of Bayesian inference algorithms. Each algorithm we present makes better use of the data and its structure. The concept of expectation maximization will be discussed. We will prove that the update rule in the last algorithm we present is exactly the update rule we would get if we would use expectation maximization. Using one year of data, we were able to reconstruct the three social networks with a maximal absolute error of 0.0013.

Finally, we will apply the model to generate sensor data and the last inference algorithm to an example of an office building with twelve occupants and we analyze the results. Using one year of data, we were able to reconstruct the social network with a maximal absolute error of 0.0001 for this example.

Keywords: interaction, occupants, sensor data, maximal independent set, Bayesian inference, expectation maximization

Contents

1 Introduction

Recently, Philips has implemented Connected Lighting in The Edge, an office building in Amstelveen. By equipping lights in an office building with sensors, and connecting those sensors to one another, one gets a very dense sensor network. Using the information that can be extracted from this network, the energy consumption in the building can be reduced and the available space can be used more efficiently. A relevant question is what extra information can we extract from the sensor network. The goal of this project is to infer a social network of interaction in an office building based on sensor data from lighting systems.

Obtaining sensor data
Two complications arise when it comes to obaining sensor data. Since this data is classified as sensitive, we have no access to real sensor data. Moreover, we want to have control over the input for the generation of sensor data. In this way, we can use the input as benchmark to evalute the perfomance of the inference. Therefore, the first goal of this project is to develop a model to generate sensor data in which we can use interactions amongst occupants as input. We will first study literature on the simulation of occupancy in office buildings. This is a major part of the research on the simulation of energy consumption in office buildings. We will discuss different approaches to the simulation of occupancy, which are described in [15], [13],[9] and [14], and we will discuss the relevance of this literature for this thesis. Next, we will present a model to generate sensor data in an office building which uses the interaction amongst occupants as input. We will present a method to simulate interactions, i.e. occupants meeting each other. Within this method, we will need to determine all maximal independent sets of a graph. An algorithm that is able to do this, called the Bron-Kerbosch algorithm [1], will be discussed. The interactions will be used to simulate movement of the occupants in the office building. The movements will be used to generate the sensor data.

Data analysis
After we have generated data using the model we have developed, we will analyze the data in order to infer the social network. First we will introduce three examples of an office building with six occupants. These examples will be used to evaluate the performance of the inference. Next, we will discuss the main challenge for the inference. This challenge arises from the fact that information is lost when the movements are transformed into sensor data. We will propose a series of algorithms, based on the concept of Bayesian inference. These algorithms will each make better use of the data and its structure. The concept of expectation maximization [4] will be discussed. This concept is used for parameter estimation in processes where there is both observed and unobserved data. We will prove that the update rule for the parameter estimation in the last algorithm we present is the update rule we would use if we would apply the concept of expectation maximization to our inference problem.

Outline of the thesis:
In chapter 2 we will present details that are imporant throughout this thesis. In chapter 3 we will discuss the literature on occupancy simulation, the Bron-Kerbosch algorithm and the concept of expectation maximization. In chapter 4 we will introduce our model to generate sensor data. In chapter 5 we will present the three examples that will be used to test the inference methods. In chapter 6 we will present a series of inference algorithms. In chapter 7 we will apply our model to generate sensor data and the last inference algorithm to an example of an office building with twelve occupants. Chapter 8 contains a summary and discussion of the previous chapters. Moreover, we will formulate several ideas and recommandations for further research in this chapter.

2 Problem description

In this chapter we present several details of the problem that are important throughout the project. Also we present a flowchart of the various concepts of this thesis.

The goal of this project is to infer a social network using sensor data from lighting systems. The first part of this project is devoted to generating a set of sensor data. By modeling an office space, the movement of the occupants and the interaction between the occupants, we can simulate the detailed locations of the occupants. By then defining the properties of the sensors that are build into the lighting system, the location data can be transformed into sensor data. This sensor data will then be analyzed to infer the social network of the occupants.

Modeling an office space
With regard to modeling the office space, i.e. the framework, the following points are important to consider:

- What kind of offices will there be?
  Modern office buildings are equipped with (a mixture of) the following kinds of offices: personal offices, shared offices and open offices. For the examples we will consider, we will only have personal offices (one office per occupant).

- Will there also be other rooms in the building?
  One option is to let all activity take place in the personal offices. Alternatively, one may add rooms that have a specific function, e.g. meeting rooms. In the examples we will use, we will add meeting rooms that will be used for meetings.

Modeling the movement of occupants
When we have a proper framework in which the occupants will move, we need to think about the following questions regarding the movement of the occupants:

- Will we consider a varying speed with which the occupant is moving through the office space?
  The most straightforward would be to let all occupants move with the same constant speed between the current location and destination. However, it may be more realistic to let the speed of each occupant be different, since in reality some people walk faster than others. Also, it would be realistic to let all occupants move with varying speeds. In our model to generate sensor data, we choose to to let all occupants move with the same constant speed.

- Will we allow occupants to make detours when moving from one room to the other?
  The most straightforward would be to not allow occupants to make detours. This means that occupants will always follow the shortest path between the current location and destination. In reality, there may be reasons for occupants to make detours. However, in our model we will ensure that all occupants do not move more than is neccessary to move from the current location to destination.

Modeling interaction between occupants
Since we want to infer a social network of the occupants, we need to have a close look at how we model the interaction between occupants. Some important points with regard to interaction are:

- How will we define the interaction between occupants?
  The simplest definition of interaction is to say that two occupants are interacting if they are in the same room. One may also say that interaction, or at least the interaction we are interested in, is more complex. For example, we may say that two occupants can only be interacting in one of the meeting rooms and not in the personal offices. We will define interaction in this project in accordance with the first definition of interaction we mentioned.

- How will we display the amount of interaction between occupants?
  One could choose to denote interaction by a binary variable for each pair of occupants (where a 1 means that a pair of occupants interacts frequently). Alternatively, we can say that the interaction for a pair of occupants is the fraction of time they are in the same room, such that it can be

represented by a number from the interval $[0, 1]$. We choose to adopt the second way to denote interaction. Hence, the interaction amongst the occupants can be represented by a graph where the occupants are represented by vertices. The edge weights of this graph will be the fractions of time that the occupants in a pair interact. This graph will be called the social network or interaction graph

Transforming location data into sensor data
Eventhough the transformation of location data into sensor data seems straightforward, the following questions are worth considering:

- What kind of sensor will be used?
  One could choose to use a simple infrared motion detector. An alternative would be to use a more advanced heat sensor that is able to count the number of occupants within its reach. We choose to use the simple infrared sensor since this is the most common kind of sensor in office buildings. Moreover, we want to see in this project how much we can extract from the simplest kind of sensor, i.e. the least informative kind of data.

- Will the sensors be equipped with its own accuracy?
  The simplest approach would be to let all sensors work completely accurate. With regard to realism however, one may choose to implement the possibility to have failures occuring in the sensor network. For this project, we choose to let our sensor network work perfectly, i.e. an accuracy of 100% for all sensors.

Analyzing the data
After the data is generated, we can focus on the inference. Two important aspect of the data analysis and inference are:

- What characteristics of the social network will be infered?
  Since the input for the model to generate the sensor data will be a probability of interaction (number in $[0, 1]$) for each pair, one can choose to estimate these numbers in the inference. In this way, we would attempt to reconstruct the detailed input. Alternatively, one could choose to infer less detailed characteristics of the social network, e.g. clusters. We choose for this project to infer the complete social network, i.e. the interaction probabilities for all pairs of occupants.

- How can we evaluate the performance of the inference, i.e. what are appropriate ways to compare the infered network to the social network that was put into the model?
  Since the input for the model and the result of the inference will be in the same format, we can easily evaluate the performance of the inference. The two main quantities we will calculate to compare the input to the results are the average and maximal absolute error between the realized fractions of interaction and the interaction probabilities that are obtained in the inference.

Now that we have discussed the details of various aspects of this project, we will look at a flowchart of this project in which we can see how various concepts we will discuss in this thesis are connected. This flowchart can be found in figure 1.

In section 4.1 we will present the representation of the office building, i.e. the framework. The movement mechanism will be introduced in section 4.2. Section 4.3 is devoted to the generation of a meeting schedule based on the input of the model (social network). In section 4.4 we explain how this meeting schedule, together with the framework and the movement mechanism, can be transformed into detailed location data. In section 4.4 we will also show the transformation of the detailed location data into sensor data.

In chapter 6 we will infer a probability distribution of locations based on the sensor data. The wide arrow in the flowchart indicates that, in contrast to the other steps in the flowchart, this step yields an estimation procedure. We will use this probability distribution in chapter 6 to construct the estimated social network.

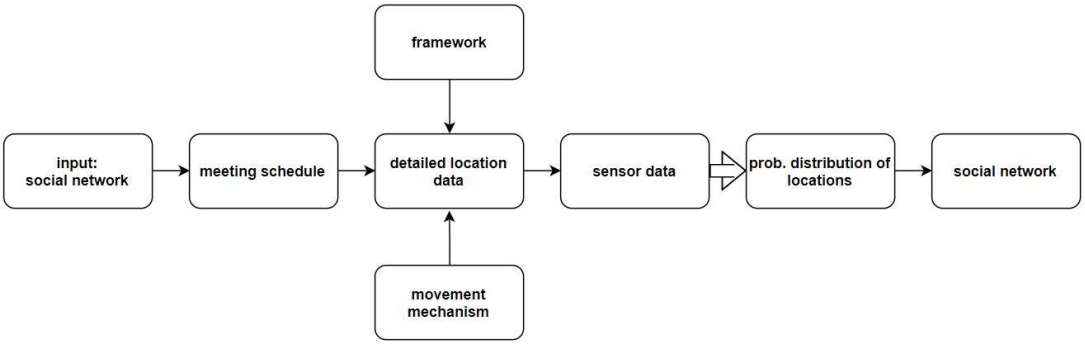

Figure 1: Flowchart of the generation of sensor data and the inference of the social network

3 Literature review

In this chapter we will discuss literature on the simulation of occupancy in office buildings and the relevance of this literature for our project. Next, we will discuss the Bron-Kerbosch algorithm. We will conclude this chapter with an introduction to the concept of expectation maximization.

3.1 Literature on simulation of occupancy in office buildings

It has been established that the occupancy of a building has a large impact on its energy consumption. For example, the understanding of the occupancy of a building allows engineers to design a lighting system that can reduce the amount of energy needed to illuminate the building. The existing models for simulation of occupancy in buildings may be very useful for the model that will be developed for this project. The literature distinguishes four levels of occupancy based on the scale and the level of detail (see [6]):

1. the number of occupants present in a building at a particular time
2. the occupancy status of a space at a particular time, i.e. whether a space is occupied by at least one person
3. the number of occupants present in a space at a particular time
4. the location of an occupant in a building at a particular time

For each of these levels, models have been developed to simulate occupancy. For the three more detailed levels of occupancy, we will discuss some models that together describe the most common approaches to occupancy simulation.

3.1.1 Occupancy status of a space

Yu [15] uses a genetic programming algorithm to learn the behaviour of an occupant in a single-person office by using sensor data. Using six variables, the genetic program learns the occupancy rules. The prediction accuracy of the algorithm is as high as 80-83% using test data from five different offices. Although the results are very promising, the major drawback of this approach is the neccesity to have data which is not the case for this project. Also, the fact that this model only works for single-person offices limits the possibility to apply this model.

Wang et al [13] examined the statistical properties of occupancy for single-person offices in a large office building. They found that vacancy intervals (time intervals during which an office is unoccupied) are exponentially distributed with a constant coefficient. Occupancy intervals (time intervals during which an office is occupied) were found to be more complex to model since their distribution is varying over time. This model is useful to simulate sensor data for a single-person office. However, this model may lack the details to be incorporated into a simulation of an entire building, since the occupancy in the different offices of a building is complex due to interdependency. Moreover, the limitation to single-person offices is again a barrier for using this model since there is no information about the modeling for rooms that are occupied by more people, e.g. meeting rooms.

3.1.2 Number of occupants present in a space

Page et al. [9] propose a model that uses inhomogenous Markov chains to model generate a time series of the state of presence of each occupant in a zone, e.g. a particular office. They even incorporated a mechanism to simulate longer periods of absence for the occupants. The main advantages of this model are the easy implementation and the limitless scalability of the model. However, it will again be a challenge to model the interdependency between the various offices of the office building.

3.1.3 Location of an occupant in a building

Wang et al. [14] propose a model in which homogenous Markov chains are used to simulate the location of occupants. This model has an advantage over the other models since the interdependency between the various offices is dealt with in this model, i.e. an occupant will be at exactly one place at any moment. Another property of this model is that it provides the possibility of modeling interaction. A drawback of this model is the large number of matrices that need to be stored since we need a transition probability matrix for each of the occupants. The size of the transition matrices may also be a problem if the office building is too large.

3.1.4 Relevance of the models for simulation of occupancy for this project

None of the models that we mentioned take into account the actual movement of an occupant. This means that an occupant moves from one location to the next within one time step, enducing a lack of detail with regard to the actual location of the occupants. Therefore, we need to think of a movement mechanism for the occupants such that the location data of the occupants is more detailed and can be translated into detailed sensor data. Also, apart from the last model we discussed, these models do not allow the modeling of interaction. For the last model we have that some sort of interaction can be modeled within the choice of the transition probabilities for all transition probability matrices. However, it is not clear whether this can be done with the level of detail we need for this project. Therefore, we need to find a modeling approach in which we can model interaction more explicitly.

3.2 The Bron-Kerbosch algorithm

In order to generate a meeting schedule, we will need to determine all maximal independent sets of a particular graph. Therefore we will look at the complement of the graph. In this way, determining all maximal independent sets in the original graph can be achieved by finding all maximal cliques in the complementary graph. This enables us to use the Bron-Kerbosch algorithm which is an algorithm that can find all maximal cliques of a graph. Even though there exist other algorithms that are more efficient in theory (e.g. [7] and [12]), it is frequently reported that the Bron-Kerbosch algorithm (and the improved versions of the original algorithm) are more efficient than the alternatives in practice, see [3].

The basic form of the algorithm, see [1], is a recursive backtracking algorithm that finds all maximal cliques in a given graph. Let R denote the temporary result, P the set of possible candidates and X the set of excluded vertices. On the first call to the algorithm we have that $R = X = \emptyset$ and P is the vertex set of the graph. Moreover let $N(v)$ denote the set of neighbors of the vertex v. The algorithm can be described as follows:

Choose a vertex v from the set P to expand the temporary clique R into $R \cup v$. Then remove the non-neighbors of v from P and X. Next, choose another vertex from P to expand the temporary clique. Once P is empty, we report R as a maximal clique if X is empty as well. If X is not empty, then R contains a subset of a clique that was already found. The next step is to backtrack to the last chosen vertex, restore P, R and X as they were before this vertex was added. Remove this vertex from P and add it to X. Then the next vertex can be chosen to expand. If P is empty, then backtrack to the superior level.

This basic form of the algorithm is inefficient if the graph contains many non-maximal cliques. Indeed, the algorithm makes a recursive call for every clique whether it is a maximal clique or not. To speed up this process, Bron and Kerbosch introduced an improved algorithm, see [1] in which a pivot vertex u is chosen from the set P. The idea is that now we do not need to consider every vertex v to expand R but just u and the non-neighbors of u. This is true because any maximal clique will either contain u or one of the non-neighbors. Later investigation ([3], [11]) showed that the pivot vertex u should be chosen from the set $P \cup X$ instead of just P.

The remaining question is what the best way is to choose the pivot vertex u. Tomita et al. [11] propose to choose the vertex u that has the highest number of neighbours in P. This seems a logical choice as this minimizes the number of points we need to consider to expand the temporary clique. The algorithm

we will use is the Bron-Kerbosch algorithm with pivoting and the pivoting strategy of Tomita et al. In pseudocode the algorithm can be formulated as:

Algorithm 0: Bron-Kerbosch/Tomita

Initialization: $R = X = \emptyset$, $P = V$
BKT(R, P, X)
if *P and X are both empty* **then**
  return R as maximal clique
end
choose the pivot vertex u in $P \cup X$ as the vertex with the highest number of neighbors in P
for *each vertex v in $P \backslash N(u)$* **do**
  BKT$(R \cup v, P \cap N(v), X \cap N(v))$
  $P \leftarrow P \backslash v$
  $X \leftarrow X \cup v$
end

In section 4.3 we will see an example of a graph for which we determine all maximal independent sets using the Bron-Kerbosch algorithm.

3.3 Expectation maximization

In order to estimate the parameters of the probability distribution of locations (with which we determine the estimated social network) we will use the concept of expectation maximization.

Suppose that we have a statistical model which, based on unknown parameters θ, generates two sets of data: observed data $\mathbf{Z}$ and unobserved data $\mathbf{S}$. In our case, θ will be the probability distribution of locations, $\mathbf{Z}$ is the sensor data and $\mathbf{S}$ will be the scheduled locations (see chapter 6). From the statistical model with which the data was generated we can derive the likelihood function $L(\theta; \mathbf{Z}, \mathbf{S}) = p(\mathbf{Z}, \mathbf{S}|\theta)$. The MLE (maximum likelihood estimation) of θ can be expressed as:

$$\hat{\theta}_{MLE} = \underset{\theta}{\operatorname{argmax}}\ L(\theta; \mathbf{Z}) = \underset{\theta}{\operatorname{argmax}} \int p(\mathbf{Z}, \mathbf{S}|\theta) d\mathbf{S} \tag{1}$$

The integral that needs to be maximized is often intractable. A very common cause for this intractability is the fact that $\mathbf{S}$ is a sequence of events so that the state space grows exponentially as the length of the sequence increases. This is the case in the process to generate sensor data as well. The expectation-maximization algorithm (EM algorithm), see [4], attempts to find the MLE of θ in a iterative way by repeating the following two steps:

- E step: Calculate the expected value of the log likelihood function, with respect to the conditional distribution of $\mathbf{S}$ given $\mathbf{Z}$ under the current estimate of the parameters $\theta^{(k)}$:

$$Q(\theta|\theta^{(k)}) = E_{\mathbf{S}|\mathbf{Z},\theta^{(k)}}(log\ L(\theta; \mathbf{Z}, \mathbf{S})) \tag{2}$$

- M step: Find the parameters that maximize $Q(\theta|\theta^{(k)})$. Hence:

$$\theta^{(k+1)} = \underset{\theta}{\operatorname{argmax}}\ Q(\theta|\theta^{(k)}) \tag{3}$$

Depending on the starting values $\theta^{(0)}$, this algorithm may converge to a local maximum instead of the global maximum. For the best possible results, using available information to determine the starting values is recommended. There are numerous applications for the concept of expectation maximization, such as:

- estimation of parameters for image segmentation e.g. [2]
- estimation of the parameters of a hidden Markov model: the Baum-Welch algorithm ([10])

In section 6.6 we will see how the concept of expectation maximization is applied to find the probability distribution of locations.

4 A model to generate sensor data

In this chapter we will discuss how a set of detailed location data can be simulated. We will then discuss how this can be transformed into sensor data. We will also show the various steps of the sensor data generation by means of an example. This example will be an office building with three personal offices and one meeting room. The office buildng has three occupants.

4.1 Modeling an office space

For this component of the data generation it was chosen to take a simple approach. We represent the office building by a grid in which each cell can represent a work area (a personal office or a meeting room) or a segment in one or more paths which connect all the work areas. For the example we consider in this chapter, the grid can be found in figure 2.

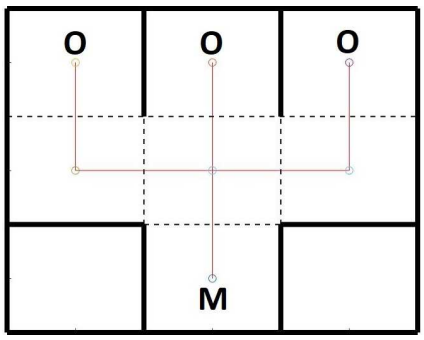

Figure 2: The office building, where the three personal offices are marked with the symbol **O**, the meeting room is marked with the symbol **M** and the other cells form the paths between the different work areas. The bold black lines are the walls of the building.

4.2 Modeling movement of the occupants

In the model we will make use of two time scales. The first will be called a time step, which is indexed by t, and is a short amount of time, e.g. second. The second is called a time period, which is indexed by τ, and is a block of time steps, e.g. half an hour.

Using the interaction model, which will be described in detail in the next section, we have that each occupant is assigned a location (work area) at every time period. A sequence of grid cells, representing the shortest path between the current work area and the destination, is generated for each occupant. At the beginning of the time period, occupants will move from their current work area to their assigned work area by moving along the grid cells in the sequence with the same constant speed (which can be expressed as the number of time steps it takes to go from the center of a grid cell to the center of the next grid cell) until the assigned work area is reached. Once the occupant has arrived at the new work area, he will stay there for the remainder of the time period.

In section 4.4 we will show an example of how this movement mechanism is used to generate detailed location data.

4.3 Modeling interaction amongst the occupants

The main feature of the model is the explicit modeling of interactions. Therefore, this section will contain more information compared to the other sections in this chapter.

We define the amount of interaction between two occupants as the fraction of time periods during which the two occupants are scheduled to be at the same work area. Hence, the amount of interaction can

be represented by a number in the interval [0,1]. Now suppose that we have a desired probability of interaction for each pair of occupants. For the example of this chapter, these interactions are displayed in the graph in figure 3.

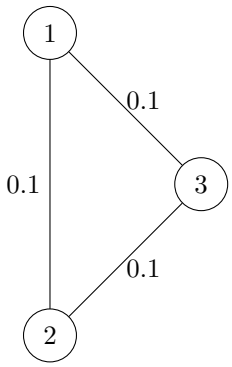

Figure 3: The interaction graph for our example

The following model will allow us to incorporate these probabilities of interaction into a mechanism that assignes a work area to each occcupant at every time period:

Consider a bipartite graph $G(V, E)$ where the vertex set V is partioned into the sets $U = \{u_n,\ n = 1, ..., N\}$ (representing the N occupants of the office) and $W = \{w_m,\ m = 1, ..., M\}$ (representing the set of M possible meetings). An edge $(u_n, w_m) \in E$ if occupant n is part of meeting m. Such a graph is called an *affiliation graph*, a concept used in network modeling and network analysis. For an easy-to-read introduction to the concept of affiliation graphs, we refer to paragraph 4.3 of [5]. The affiliation graph of our example is given in figure 4.

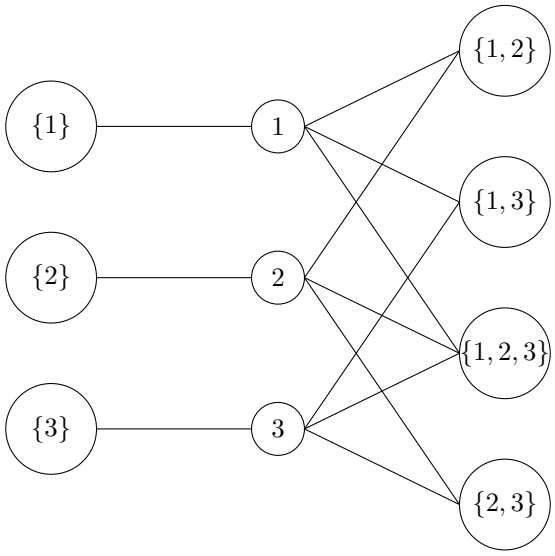

Figure 4: The affiliation graph for our example

Note that we have also a meeting for each individual occupant. Now consider the graph $G' = (W, E')$ which is the bipartite projection of the graph $G(V, E)$ onto the set W. By definition, an edge $(w_m, w_k) \in E'$ if and only if $\exists u_n \in U : (u_n, w_m), (u_n, w_k) \in E$. The interpretation of this graph is that two meetings are connected if they have one or more occupants in common (and hence can not be scheduled simultaneously). We will call this graph the meeting graph. For our example, this graph can be found in figure 5.

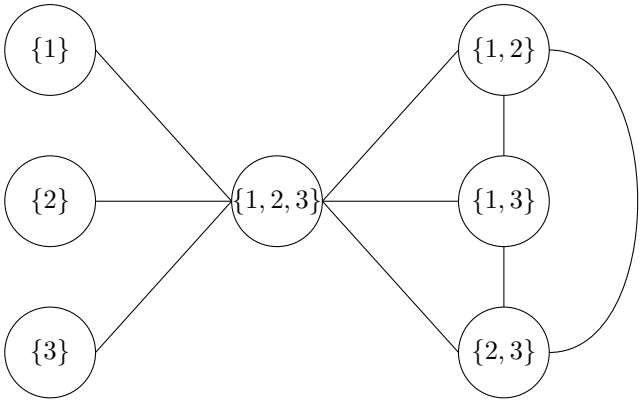

Figure 5: The meeting graph for our example

Moreover, let $\{M_l,\ l = 1, ..., L\}$ be the set of maximal independent sets in G'. A set $M_l \subseteq W$ is a maximal independent set if it has the following two properties:

1. $\forall w_m, w_k \in M_l : (w_m, w_k) \notin E'$ where $m, k \in \{1, ..., M\}$
2. $\forall w_q \in M_l^c\ \exists w_m \in M_l : (w_m, w_q) \in E'$ where $m, q \in \{1, ..., M\}$

We have that by choosing a maximal independent set of the graph G', a set of meetings is chosen such that each occupant is in a meeting and we have no occupants who are assigned two meetings at the same time. A meeting schedule can be simulated by choosing a maximal independent set from the graph G' according to a stationary probability distribution π (a probability vector of size L) at every time period. The fact that we have a meeting for each individual occupant is essential for this statement to be true.

Two questions have not been answered yet. The first is how we find all maximal independent sets. The second is how we should choose the probability distribution π.

4.3.1 Application of the Bron-Kerbosch algorithm

In order to find all maximal independent sets of the graph G' (the meeting graph), we will make use of the Bron-Kerbosch algorithm (see section 3.2).

To do so, we will look at the complement of G', which will be denoted by $G'^c(W, E'^c)$, in which a pair of meetings (w_k, w_m) is connected if they have no occupants in common and hence can be scheduled together. Let $\{C_l, l = 1, ..., L\}$ be the set of all maximal cliques in the graph G'^c. A set $C_l \subseteq W$ is a maximal clique if it has the following two properties:

1. $\forall w_m, w_k \in C_l : (w_m, w_k) \in E'^c$ where $m, k \in \{1, ..., M\}$
2. $\forall w_q \in C_l^c\ \forall w_m \in C_l : (w_m, w_q) \notin E'^c$ where $m, q \in \{1, ..., M\}$

We now have that finding all maximal independent sets of G' can be accomplished by finding all maximal cliques in the graph G'^c. For our example, the graph G'^c is given in figure 6.

If we apply the Bron-Kerbosch algoritm to the graph in figure 6, we find the maximal independent sets as can be foumd in table 1.

Maximal independent set (l)	1	2	3	4	5
Meetings	$\{1, 2, 3\}$	$\{1\}$, $\{2, 3\}$	$\{2\}$, $\{1, 3\}$	$\{3\}$, $\{1, 2\}$	$\{1\}$, $\{2\}$, $\{3\}$

Table 1: All maximal independent sets of the meeting graph

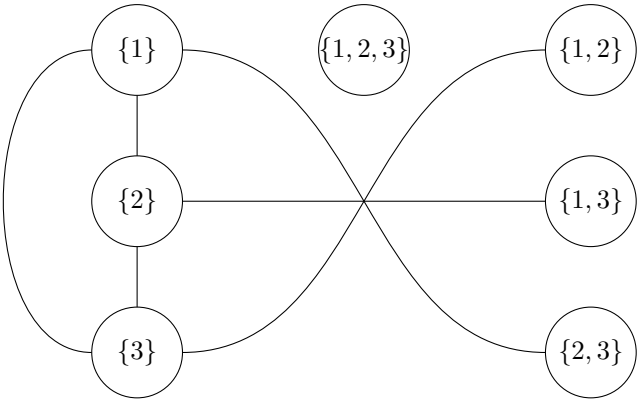

Figure 6: The complement graph G'^c for our example

4.3.2 Finding the probability distribution π

There is only one task that needs to be done before we can generate a meeting schedule. This task, determining the probability distribution π, is the main feature of the model to generate sensor data.

Given the graphs G and G' and a probability distribution π, we can construct a third graph $G'' = (U, E'')$, the bipartite projection of the graph $G(V, E)$ onto the set U. In this graph two occupants wil be connected if and only if there is a meeting to which both occupants belong. Consider the following weight function for the edges of the graph G'' $\omega : E'' = U \times U \to [0, 1]$:

$$\omega(i,j) = \sum_{l=1}^{L} \mathbb{1}\{\exists w_m \in M_l : \ (u_i, w_m), (u_j, w_m) \in E\} \cdot \pi_l \quad i, j \in \{1, ..., n\} \quad i \neq j \tag{4}$$

The interpretation of this weight function is rather elegant: the weight of an edge equals the probability that these two neighbouring occupants in the graph will be in a meeting together and hence will be interacting.

The following question arises: How should we choose the probability distribution π? One possibility is to use the relation between π and the edge weigths in the graph G''. More precisely, we may choose π in such a way that the resulting edge weigths are close to desired values. To do so, define the matrix $A \in \{0, 1\}^{R \times L}$, where $R = \frac{N(N-1)}{2}$ is the number of pairs. Let $a_{r,l} = \mathbb{1}\{\exists w_m \in M_l : \ (u_i, w_m), (u_j, w_m) \in E\}$ where the index r belongs to a pair (u_i, u_j) of occupants in the office. The meaning of this matrix is that $a_{r,l} = 1$ if and only if the two occupants belonging to the pair r will get assigned the same meeting if the maximal independent set M_l is chosen. Moreover, define $p \in [0, 1]^R$ to be the vector of desired edge weights in the graph G''. Then choosing an appropriate probability distribution π can be formulated as the following constrained linear least-squares problem:

$$\begin{aligned} \underset{\pi}{\text{minimize}} \quad & \|A\pi - p\|_2^2 \\ \text{subject to} \quad & \pi_l \geq 0, \ l = 1, \ldots, L. \\ & \sum_{l=1}^{L} \pi_l = 1. \end{aligned} \tag{5}$$

For our example, we can construct the matrix A using table 1 and figure 4. The pairs have the following order: pair 1: $\{1, 2\}$, pair 2 : $\{1, 3\}$, pair 3: $\{2, 3\}$. The three rows of the matrix A belong to the corresponding three pairs. Figure 7 shows the matrix A for our example.

The vector p can be extracted from the interaction graph in figure 3 and is equal to $(0.1, 0.1, 0.1)^T$.

$$A = \begin{pmatrix} 1 & 0 & 0 & 1 & 0 \\ 1 & 0 & 1 & 0 & 0 \\ 1 & 1 & 0 & 0 & 0 \end{pmatrix}$$

Figure 7: The matrix A for our example

In most cases, the problem in (5) is underdetermined, meaning that there may multiple probability distributions such that the objective function of the minimization problem is equal to 0. The reason for this is the fact that the number of maximal independent sets is much larger than the number of pairs in most cases. So, one may choose to redefine the problem into the following form:

$$\begin{aligned} \underset{\pi}{\text{minimize}} \quad & f(\pi) \\ \text{subject to} \quad & A\pi = p \\ & \pi_l \geq 0,\ l = 1, \ldots, L. \\ & \sum_{l=1}^{L} \pi_l = 1. \end{aligned} \tag{6}$$

The objective function $f(\pi)$ can be chosen in several ways. One may use the objective function to ensure that the resulting probability distribution has certain properties, which will be portrayed in the meeting schedule, such as:

- minimization/maximization of the number of meetings (with more than one occupant) occuring at the same time
- minimization/maximization of the scheduling of larger meetings (meetings with more than two occupants)

One can also add constraints to the optimization problem to manipulate the probability distribution. If the latter optimization problem turns out to have no feasible solution, one can resort to the former to find an appropriate probability distribution π.

In chapters 5 and 6, we will make use of the first optimization problem. In chapter 7, we will explore the possibilities of the second optimization problem.

4.3.3 Generating the meeting schedule

At this point we are able to generate a meeting schedule. This means we will sample a maximal independent set of the graph G' at every time period according to the probability distribution π. Suppose that one time period represents half an hour and that each working day consists of ten hours. Moreover, assume that each year consists of 240 regular working days. Then building a meeting schedule for an entire year would result in a sample of size 4800, which we have chosen as the base sample size in our simulations. In chapter 6 we will only use the base sample size. In chapter 7 we will use the sample sizes as can be found in table 2.

The realized fractions of interaction in the sample can deviate from the interaction probabilities in the interaction graph. This is due to the finite length of the sample. To see the effect of the size of the sample on the difference between realized interactions and desired interactions, we look at table 2.

From this table we can see that there is a small difference between realized interactions and the interaction probabilities from the interaction graph. Since the differences are not too large, we can use samples of these lengths such that the interaction probabilities from the interaction graph are accurately represented in the meeting schedule.

	pair 1	pair 2	pair 3
Interaction probability	0.1000	0.1000	0.1000
T=1200 (3 months)	0.0983	0.1067	0.1000
T=2400 (6 months)	0.1017	0.1038	0.0958
T=4800 (1 year)	0.1015	0.1040	0.1025
T=9600 (2 years)	0.0995	0.1032	0.0981

Table 2: Interaction probabilities and the realized interactions for different sample sizes

4.4 Generation of location data and transformation into sensor data

Using the layout of the office building (see figure 2) and the set of meetings taking place during a particular time period, we can assign a work area to each of the occupants at every time period. By using the movement mechanism as described in section 4.2, we have the location of each of the occupants at every time step, i.e. detailed location data. Hence, we now have a mechanism to generate detailed location data in which interaction probabilites can be incorporated.

Consider as an example the case in which all occupants are in their personal office and are scheduled to meet each other in the meeting room. Moreover, assume that the number of time steps to get from one grid cell to the next is 1. The detailed location data of all occupants moving from their personal offices to the meeting room can be found in the left column of figure 8.

We have chosen to take a rather simple approach for the transformation of the location data into sensor data. Our sensor network will consist of one sensor per grid cell. At every time step, the sensor will detect whether there is at least one occupant within the grid cell, in which case it will send a value 1, or whether there is no occupant within the grid cell, in which case it will send a value 0. Hence, we translate the location data set, which consists of the number of a grid cell for each occupant at every time step, into sensor data consisting of either a 0 or 1 for every grid cell at every time step. We also have chosen to let our sensors be ideal in the sense that the accuracy of each sensor in the network is 100%. The sensor data that corresponds to the detailed location data (which is the left column of figure 8) can be found in the right column of figure 8.

One way to present the detailed location data is by the matrix X where $x_{t,n}$ is the number of the grid cell where occupant n is at time step t. This data can be transformed into the matrix Y where $y_{t,q}$, with q being the index of the sensor/grid cell, is the value of sensor q at time t, or equivalently: $y_{t,q} = \mathbb{1}\{\exists x \in x_{t,1:n} : x = q\}$.

4.5 Scalability of the model

One very important aspect of the model that we have not discussed yet is the scalability of the model. Therefore we will look at the worst-case scenario with regard to the number of maximal independent sets of the meeting graph.

Suppose that we have an office building with N occupants. Moreover, assume that the interaction graph is a complete graph, i.e. each occupant will interact with all the other occupants. We also know that, in the worst-case scenario, the number of meetings is equal to the number of non-empty subsets of the set of occupants. The number of maximal independent sets in the meeting graph can also be derived from the fact that we consider the worst-case scenario. This will be equal to the number of partitions of the set of occupants. For an office with N occupants, the number of partitions of the set of occupants is equal to the N'th Bell number B_N. This quantity can be calculated by the following formula:

$$B_N = \frac{1}{e} \sum_{k=0}^{\infty} \frac{k^N}{k!} \tag{7}$$

For three occupants we have that the Bell number $B_3 = 5$ whereas for twelve occupants we have that the Bell number $B_{12} = 4,213,597$. If one would consider a case with twenty occupants, the number of

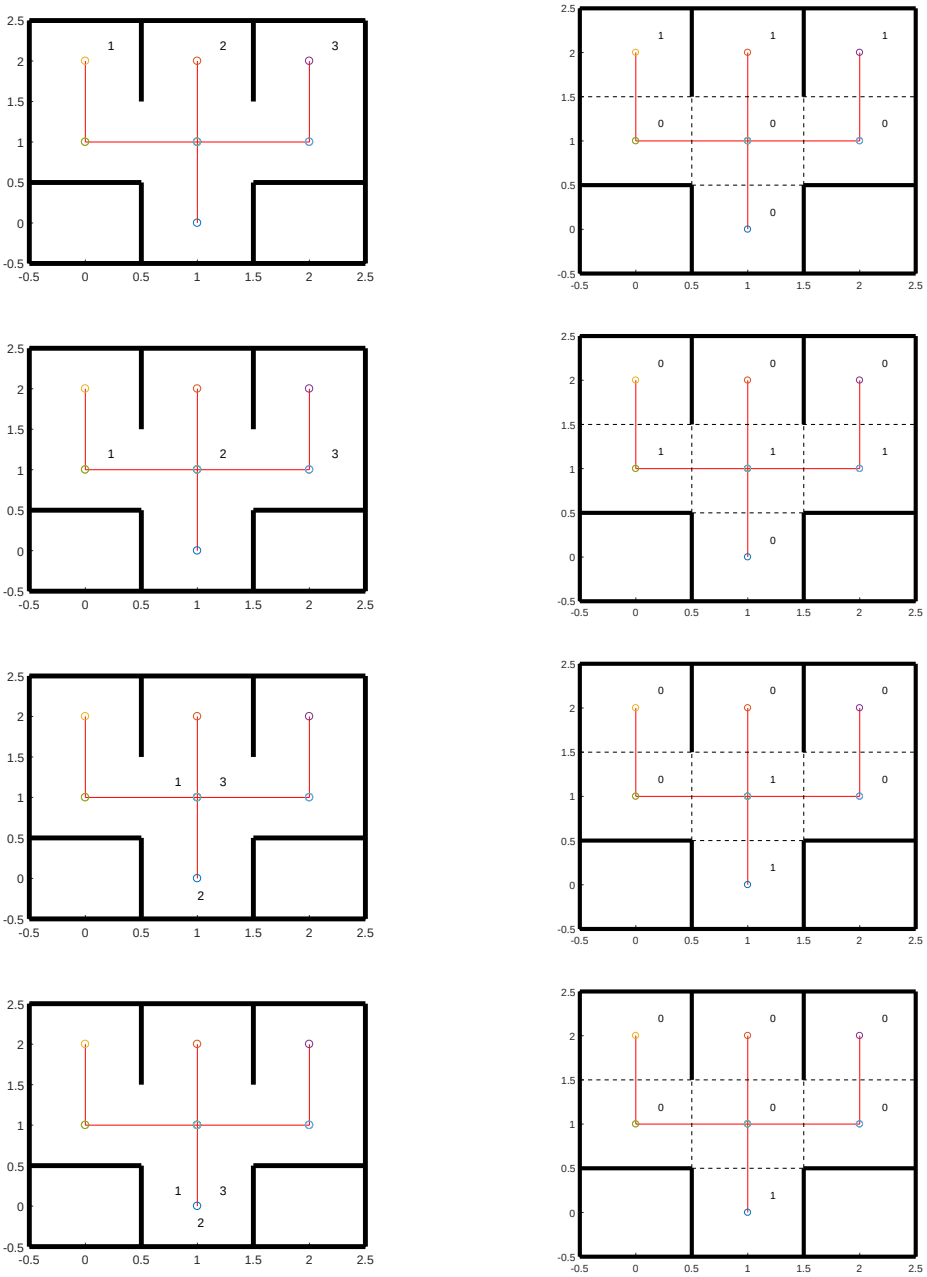

Figure 8: An example of detailed location data which is transformed into sensor data

maximal independent sets will be $B_{20} = 682,076,806,159$ in the worst-case scenario.

We may conclude that scalability is an issue for the model. However, we note that in this section we assumed the worst-case scenario. One can imagine that in practice, and in particular in larger office buildings, the interaction graph will not be complete. Moreover, the number of meetings can be limited manually. An example in which one may decide to exclude particular meetings is the following: *Suppose that in an office with a large number of occupants, there is a meeting to which all occupants belong. One can imagine that the meeting to which all occupants belong except for one particular occupant will not exist.* Nonetheless, scalability is an issue for the application of the model to larger office buildings.

5 Three examples of an office building with six occupants

In this chapter we will discuss the three examples that we will use to evaluate the performance of the various inference methods (which will be presented in chapter 6). All three examples will be examples of an office building with six occupants. These three examples together will give insight in the influence of two aspects of the interaction graph (magnitude of interaction probabilities and the density of the graph) on the performance of the inference.

5.1 The office space

All three examples will use the same office space, i.e. the framework. This framework can be found in figure 9.

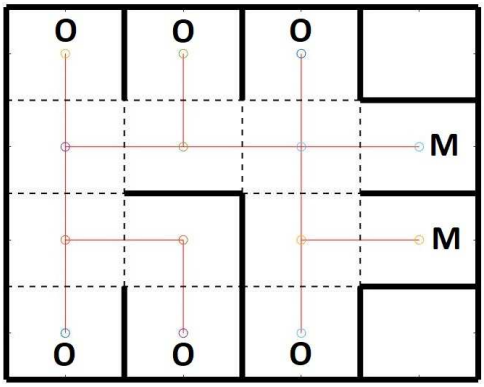

Figure 9: The office building, where the six personal offices are marked with the symbol **O**, the two meeting rooms are marked with the symbol **M** and the other cells form the paths between the different work areas. The bold black lines are the walls of the building.

5.2 Example 1

Example 1 will be the base case. The interaction graph for this example can be found in figure 10.

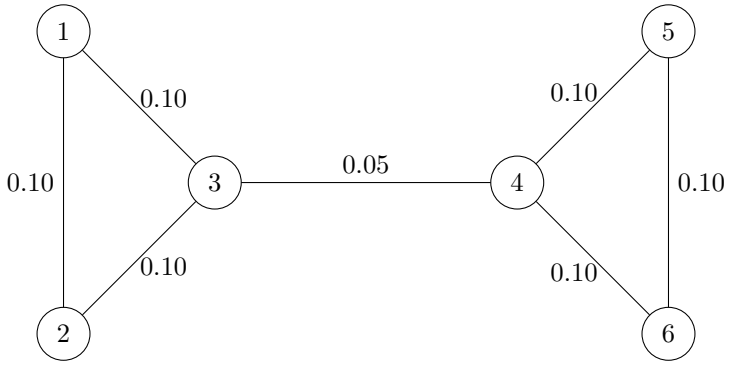

Figure 10: The interaction graph of example 1

Next, we need to define all possible meetings that can occur. For this example, it was chosen to let the set of possible meetings equal the set of all completely connected subsets of vertices in the interaction graph. Besides a meeting for each individual occupant, there will be a meeting for each connected pair (seven meetings) and a meeting for each triangle in the interaction graph (two meetings), meaning that there will be fifteen possible meetings (see table 3).

Meeting number	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
Participants	1	2	3	4	5	6	1,2	1,3	2,3	1,2,3	3,4	4,5	4,6	5,6	4,5,6

Table 3: Set of meetings

5.3 Example 2

This example was chosen to evaluate the effect of the magnitude of the interaction probabilities on the performance of the inference. Example 2 is the same as example 1 except for the fact that the interaction probabilities are multiplied by a factor of three. The interaction graph of example 2 can be found in figure 11. The set of meetings for this example can be found in table 3

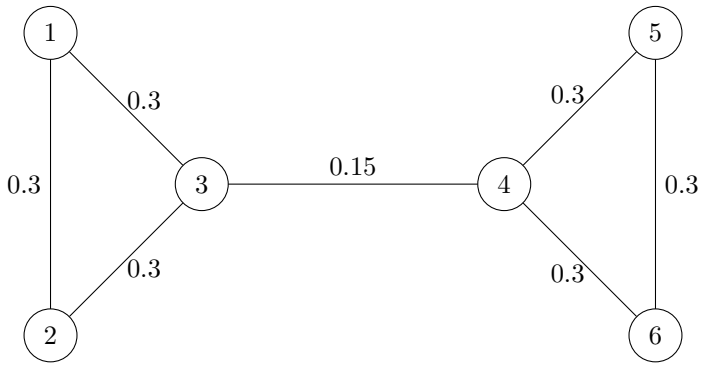

Figure 11: The interaction graph of example 2

5.4 Example 3

The third example uses the same office building for the six occupants but has a more dense interaction graph which can be found in table 4.

Occupant	1	2	3	4	5	6
1	0	0.10	0.10	0.05	0.05	0.05
2	0.10	0	0.10	0.05	0.05	0.05
3	0.10	0.10	0	0.05	0.05	0.05
4	0.05	0.05	0.05	0	0.10	0.10
5	0.05	0.05	0.05	0.10	0	0.10
6	0.05	0.05	0.05	0.10	0.10	0

Table 4: Interaction matrix for example 3

For the groups $\{1, 2, 3\}$ and $\{4, 5, 6\}$ we have that every occupant interacts with all other occupants in the same group with probability 0.10. Also, each occupant in a particular group will interact with each of the members of the other group with probability 0.05. This example also has a larger set of possible meetings. Besides the meetings that already exist for examples 1 and 2, we have added the meetings that can be found in table 5.

Meeting number	16	17	18	19	20	21	22	23
Participants	3,5	3,6	1,4	2,4	1,5	1,6	2,5	2,6

Table 5: Extra meetings for example 3

5.5 Location allocation rules

Next, we will determine the location of each meeting that happens during a particular time period, i.e. assign a personal office or a meeting room, according to the following allocation rules:

- All meetings with only one participant (working individually) will take place in the personal office of that participant.
- All meetings that have more participants will take place in one of the two meeting rooms (if there are no more than two of those meetings during that time period).
- If there are three meetings with more than one participant, the following allocation rules apply:
  - If occupant 3 is involved, his personal office will be used. The same holds for occupant 4
  - If occupants 3 and 4 are meeting together, the office of occupant 3 will be chosen.
  - If neither occupant 3 or occupant 4 is part of the meeting, the meeting will take place in a meeting room.
- All the meetings that will take place in one of the meeting rooms will be assigned a meeting room at random.

Suppose, for illustrative purposes, that the meetings 3, 4, 7 and 14 are sampled at a particular time period. The corresponding partition of the occupants is $\{\{3\},\{4\},\{1,2\},\{5,6\}\}$. Occupants 3 and 4 will be assigned their own offices. Occupants 1 and 2 will be assigned either of the two meeting rooms with probability 0.5 and suppose they get assigned the bottom meeting room. The remaining meeting room will be given to occupants 5 and 6. An illustration of this location allocation is given in figure 12.

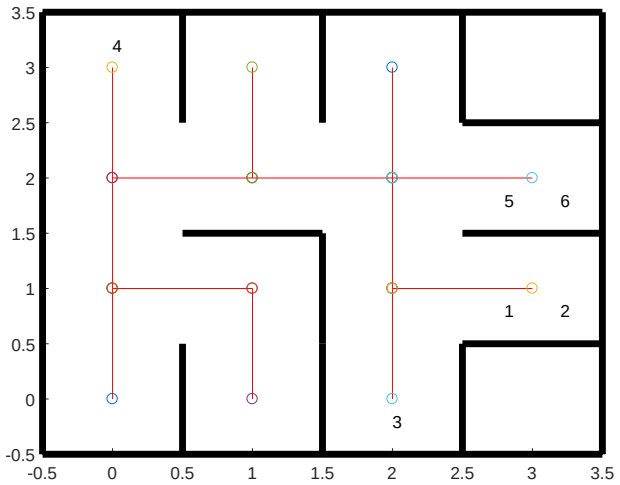

Figure 12: An example of location allocation

6 Inference of the social network

In this chapter we first discuss the main challenge for the inference. Next, we propose a series of Bayesian inference algorithms that will infer a stationary probability distribution over all possible location vectors. We use this stationary probability distribution to estimate the social network. Each algorithm we present makes better use of the data and its structure. We will prove that the update rule in the last algorithm we present is exactly the update rule we would get if we would use expectation maximization. Each of the presented algorithms will be applied to the three examples as presented in chapter 5 to evaluate the performance. Lastly, we discuss how the scalability issue as discussed in section 4.5 also plays a role in the inference.

6.1 The concept of ambiguity

The fact that we only have sensor data instead of location data raises the main challenge for the inference. One consequence of the way location data is transformed into sensor data is the fact that there exist multiple examples of location data that all result in the same sensor data. This loss of information, in which distinct examples of location data can not be distinguished based on sensor data, will be called ambiguity.

Consider as an example the case in which occupants 3 and 4 are in their respective offices and the other four occupants are in a meeting room. If we look at either the interaction graph, see figure 10, or the set of possible meetings, see table 3, we see that this can only happen if occupants 1 and 2 are in one meeting room and occupants 5 and 6 are in the other meeting room. However, when we analyze the data we do not know either the interaction graph or the set of possible meetings. Hence we have to assume that all combinations, in which occupants 3 and 4 are in their respective offices while the other occupants are divided into pairs, are possible. A visualization of this issue is given in figure 13.

6.2 List of symbols and determining the social network from the probability distribution of locations

Each of the algorithms we present in sections 6.3 - 6.7 will give an (estimated) stationary probability distribution over all possible location vectors. A location vector is a vector that contains the number of a grid cell that corresponds to a work area for each of the occupants. The set of possible location vectors can be determined using the location allocation procedure, i.e. the location allocation procedure limits the number of possible location vectors. Let $\{S_\tau\}_{\tau=1}^T$ be the random vectors that represent the scheduled locations of the occupants in each of the time periods. Since the maximal independent set at every time period is chosen in a stationary manner and the procedure to allocate locations to meetings is also time-invariant, we have that distributions of the random vectors $\{S_\tau\}_{\tau=1}^T$ are stationary.

For convenience, we present in table 6 all symbols that we will use in the remainder of this chapter.

ρ_{ij} (with estimate $\hat{\rho}_{ij}$): fraction of time periods that the occupants i and j are in the same work area
$\mathcal{S}$: set of possible location vectors
s: element of $\mathcal{S}$
θ_s (with estimate $\hat{\theta}_s$): stationary probability of location vector s (Observe that $\sum_{s\in\mathcal{S}} \theta_s = 1$)
θ: vector containing all stationary probabilities (and hence has length $|\mathcal{S}|$)
S_τ (with realization: s_τ): random vector of locations (one per occupant) for time period $\tau = 1, ..., T$
Y_τ (with realization: y_τ): the final frame of sensor data from time period $\tau = 1, ..., T$
Z_τ (with realization: z_τ): sequence of data frames from time period $\tau = 1, ..., T$ (see section 6.4)
$\mathbf{S}$ (with realization: $\mathbf{s}$): $\mathbf{S} = (S_1, ..., S_T)$, $\mathbf{s} = (s_1, ...s_T)$
$\mathbf{Z}$ (with realization: $\mathbf{z}$): $\mathbf{Z} = (Z_1, ..., Z_T)$, $\mathbf{z} = (z_1, ..., z_T)$

Table 6: List of symbols

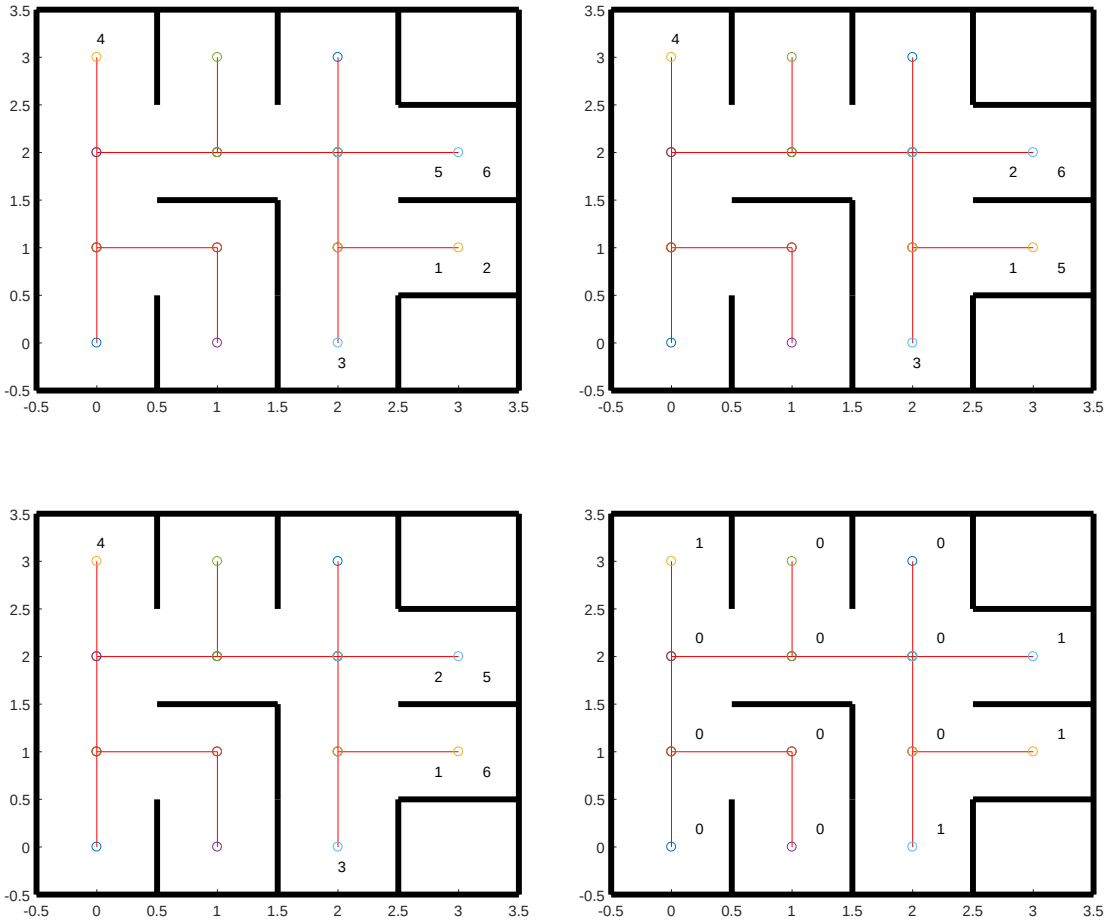

Figure 13: Three examples of location data with the corresponding sensor data

Next, we see how we can use the result of the inference algorithms to estimate the social network. Let ρ_{ij} denote the fraction of time periods that the occupants i and j are in the same work area. This fraction can be estimated by the following formula:

$$\hat{\rho}_{ij} = \sum_{s} \mathbb{1}\{\text{occupants } i \text{ and } j \text{ are in the same work area in location vector } s\} \cdot \hat{\theta}_s \tag{8}$$

We will use ρ and $\hat{\rho}$ to denote the vectors that contain the (estimated) fractions of all pairs. We will evaluate the accuracy of the estimation by calculating two measures: the average absolute error, defined as $\frac{2}{N(N-1)}||\rho - \hat{\rho}||_1$, and the maximal absolute error, defined as $||\rho - \hat{\rho}||_\infty$.

6.3 Algorithm 1

The first inference algorithm we present will use very little of the sensor data. As a consequence, ambiguity will not be resolved at all which we will prove for this algorithm.

Recall that the data we get from the sensors can be represented by the matrix Y where $y_{t,q} = 1$ if grid cell q is occupied at time step t and $y_{t,q} = 0$ otherwise. Hence, the state of the lighting system at time step t can be represented by the vector y_t, a vector of length Q, telling for each grid cell whether it is occupied at time step t. Also recall that the location data of the occupants can be represented by the matrix X where $x_{t,n}$ is the number of the grid cell where occupant n is at time step t. Hence, the location data of all the occupants at time step t is captured by the vector x_t, a vector of length N. Furthermore, let X_t denote the random variable that represents the location data of all the occupants at time step t and let Y_t denote the random variable that represents the state of the sensors at time step t. The probability distribution of Y_t conditioned on X_t is deterministic and is given by the following probability density function (pdf):

$$P(Y_t = y_t | X_t = x_t) = \prod_{q=1}^{Q} (y_{t,q} \cdot \mathbb{1}\{\exists x \in x_t : x = q\} + (1 - y_{t,q}) \cdot (1 - \mathbb{1}\{\exists x \in x_t : x = q\}))$$

According to the defined pdf, we have that a sensor signal y_t is the sensor signal belonging to the location vector x_t if and only if the signal for the sensors belonging to occupied spaces is 1 ($y_{t,q} = \mathbb{1}\{\exists x \in x_t : x = q\} = 1$) and the signal for sensors that belong to unoccupied spaces is 0 ($y_{t,q} = \mathbb{1}\{\exists x \in x_t : x = q\} = 0$). This pdf follows the assumption that all sensor have an accuracy of 100%. Hence we have that the value of this pdf is either 0 or 1. We choose to still present this as a pdf since the assumption of 100% accuracy may be dropped in future applications.

The idea behind the first inference algorithm is the following:
We sample the last data frame of each time period (denoted by Y_τ). Then for each of the location vectors and for each of the time periods, we calculate the probability of that particular location vector conditioned on the data frame of the time period. In this way, we get the posterior probability of each of the location vectors for each of the time periods. To estimate the stationary probabilty distribution for the location vectors, we calculate the time average.

In pseudo-code, we get the following algorithm:

Algorithm 1:

Inference$(Y, P(Y_\tau | S_\tau))$
Initialization: $\theta_s = \frac{1}{|\mathcal{S}|}\ \forall s \in \mathcal{S}$
for $\tau = 1$ *to* T **do**
  $\hat{p}_{loc}(\tau, s) = P(S_\tau = s | Y_\tau = y_\tau) = \frac{P(Y_\tau = y_\tau | S_\tau = s)\theta_s}{\sum_s P(Y_\tau = y_\tau | S_\tau = s)\theta_s}$
end
$\hat{\theta}_s = \frac{1}{T} \sum_{\tau=1}^{T} \hat{p}_{loc}(\tau, s)$

The initialization represents the fact that we have no prior knowledge and hence have to assume that all location vectors are equally likely. Before we look at the results for this algorithm, we will first

prove that this algorithm does not resolve the issue of ambiguity. In other words, we will prove that if two location vectors can not be distinguished based on sensor data, we have that they have the same estimated stationary probability if algorithm 1 is used.

Theorem 1: *If for two location vectors s and s', it holds $\forall y_t \ P(Y_t = y_t|X_t = s) = P(Y_t = y_t|X_t = s')$, we have that $\hat{\theta}_s = \hat{\theta}_{s'}$ for algorithm 1.*

Proof. Since $\forall y_t \ P(Y_t = y_t|X_t = s) = P(Y_t = y_t|X_t = s')$ we know that in particular $\forall y_\tau \ P(Y_\tau = y_\tau|S_\tau = s) = P(Y_\tau = y_\tau|S_\tau = s')$. Combining this with the fact that $\theta_s = \theta_{s'}$, we know that $\forall \tau \ \hat{p}_{loc}(\tau, s) = \hat{p}_{loc}(\tau, s')$ and hence $\hat{\theta}_s = \hat{\theta}_{s'}$ □

In table 10 (in the appendix) we see the estimated probabilities of interaction for example 1 and algorithm 1. In table 7 we can find the summarized results of algorithm 1 for the three examples. Based on these results we can conclude that anbiguity plays a significant role in the inference of the stationary probability distribution. In section 6.4 we will see how ambiguity can partially be resolved using more of the data and its structure.

6.4 Algorithm 2

The main feature of algorithm 2 is the fact that we use more data and we use the structure of the data to partially resolve ambiguity.

We consider the following two properties of the movement model:

- All occupants move with the same constant speed along the grid cells on their respective paths.
- The occupants move along the shortest sequence of grid cells from their current location to their destination.

If we know where each occupant is at the beginning of the time period and the destination of each occupant, we can determine for each occupant the sequence of grid cells that forms the path between the location at the beginning of the time period and the destination, including both endpoints. Under the assumption that the occupants need v time steps to go from one grid cell to another, we know that we do not need all data frames for every time period. By sampling the data frames at time steps $1, v+1, 2v+1, ..., kv+1$ at each time period, where k is chosen as the length of the longest possible path that can be chosen in the office building, we now have that in every next data frame from our sample, each occupant is either in the next grid cell of his sequence or at his destination.

Define Z_τ to be the random matrix which contains the series of data frames (as described above) from the beginning of time period τ. Moreover, define Z to contain the realizations $z_\tau, \tau = 1, ..., T$. Since all sensor data frames for a complete time period can be determined if we know the locations at the beginning of the time period and the destinations for that time period, we have that $P(Z_\tau = z_\tau|S_\tau = s, S_{\tau-1} = s')$ is known for all (z_τ, s, s'). We also know that $P(Z_\tau = z_\tau|S_\tau = s, S_{\tau-1} = s') \in \{0, 1\}$ (eventhough this is not essential). We can now use the following Bayesian estimation procedure to estimate the probability distribution for the scheduled locations:

Algorithm 2:

Inference$(Z, P(Z_\tau|S_\tau, S_{\tau-1}))$
Initialization: $\theta_s = \frac{1}{|\mathcal{S}|} \ \forall s \in \mathcal{S}$
for $\tau = 1$ *to* T **do**
  $\hat{p}_{loc}(\tau, s) = P(S_\tau = s|Z_1 = z_1, ..., Z_\tau = z_\tau) = \frac{\sum_{s'} P(Z_\tau=z_\tau|S_\tau=s,S_{\tau-1}=s')\theta_s \hat{p}_{loc}(\tau-1,s')}{\sum_s \sum_{s'} P(Z_\tau=z_\tau|S_\tau=s,S_{\tau-1}=s')\theta_s \hat{p}_{loc}(\tau-1,s')}$
end
$\hat{\theta}_s = \frac{1}{T} \sum_{\tau=1}^{T} \hat{p}_{loc}(\tau, s)$

We also know that $\hat{p}_{loc}(0, s)$ is equal to 1 if s is such that all occupants are in their respective offices and 0 otherwise. This is a consequence of the decision to let all occupants start in their personal offices during the data generation ($\tau = 0$). In table 7 we can see the summarized results of algorithm 2 for the three examples.

Even though the results for algorithm 2 are much better as compared to algorithm 1, we see that (especially for the case with the large interaction probabilities), we are not able to take away all the ambiguity. As an example of an ambigious situation we consider the following:

Suppose that we know that at a particular time period occupants 3 and 4 are in their respective offices, occupants 1 and 2 are in a meeting room together and occupants 5 and 6 are both in the other meeting room. Based on the sensor data at the beginning from the next time period, we can see that from each meeting room one occupant transfers to the other meeting room and occupants 3 and 4 stay in their respective offices. But since we do not have any prior knowledge with regard to the likelihood of any of these transfers, we need to assume that, within each meeting room, both occupants are equally likely to switch meeting rooms. As a result, we have that both situations, $\{\{3\}, \{4\}, \{1, 5\}, \{2, 6\}\}$ and $\{\{3\}, \{4\}, \{1, 6\}, \{2, 5\}\}$, are equally likely to happen, each having a different effect on the final result.

From the results for algorithm 2 we can see that this procedure is not accurate enough. The underlying problem which explain the discrepancies is the fact that, due to a lack of prior knowledge, we need to assume a priori that at every time period that all states are equally likely, eventhough we can see that, based on information from previous time periods, this is not the case. Hence, we will develop an algorithm in which the knowledge that we gained from the previous time periods is used to define a more suitable prior distribution for each time period.

6.5 Algorithm 3

The main feature of this algorithm is that the knowledge that we gained from the previous time periods is used to define a more suitable prior distribution for each time period.

The two Bayesian inference procedure we have considered in the previous sections of this chapter are examples of Bayesian updating. For each time period, we started with a prior distribution over the state space of all possible location vectors which was updated using the sensor data, the conditional probability of occurence of the sensor data (and the posterior distribution of the scheduled locations at the previous time period). The updated probability distribution, or posterior distribution, was then used to estimate the quantities of interest. Although this is a widely used and very effective approach of updating a prior belief using the information gained from data, there is one drawback to this approach if we have a closer look to our application.

As has been stated earlier, due to a lack of prior knowledge, our prior distributions for each of the time periods are not informative. However, we also know that the maximal independent sets at each of the time periods are all sampled from the same stationary probability distribution. Given the fact that the procedure to allocate work areas to meetings is also the same for every time period, we can say that the random variables S_τ also have a stationary property. Hence, we have that our prior belief for a particular time period can be improved by using what we have seen in the previous time periods. The rationale behind this statement is the following:

Suppose that, based on sensor data, we can exactly determine the locations of all the occupants during the first ten time periods. If we then consider the prior distribution for the eleventh period, it would make sense that the location vectors as detected during the first ten time periods are more likely to occur than the ones that were not detected. Therefore, we wish to define our prior for the eleventh time period in such a way that the location vectors that have been detected are more likely than the ones that were not. If we were not able to exactly detect the location vectors of the first ten time periods, we still know that the posterior distributions for these ten time periods are useful to update our prior belief for the eleventh time period even before we consider the sensor data from the eleventh time period.

One way of updating prior belief as new information becomes available is *sequential Bayesian updating*. Suppose, for illustrative purposes, that we have a discrete random variable X of which we want to estimate the probabilities $P(X = x)\ \forall x$ of which we only have a prior belief $P^0(X = x)$. Also suppose that have a discrete random variable Y that is dependent on X and that $Y|X$ is known. Lastly, suppose that we have samples $y_1, y_2, ..., y_n$. Then we can estimate the desired probabilities sequentially in the following way:

SequentialUpdating$((y_1, ..., y_n), P(Y|X), P^0(X = x))$
$p(x) = P^0(X = x)$
for $k = 1$ *to* n **do**
  $P^k(X = x) = P(X = x|Y = y_k, p(x)) = \frac{P(Y=y_k|X=x)p(x)}{\sum_x P(Y=y_k|X=x)p(x)}$
  $p(x) = P^k(X = x)$
end
return p

In words, we have that starting with a prior distribution (P^0), the prior distribution is updated after each new sample and is replaced by the posterior distribution. After all samples are used to update the distribution, the final result will be returned.

Even though this procedure is widely used and very effective in many applications, this procedure is not directly applicable to the problem that we are trying to solve. An example that will show why this is true is the following: Suppose that, based on sensor data, we can see that all occupants are in their respectives offices at a particular time period. Hence, the posterior probability of finding a particular location vector is 1 if it belongs to the location vector such that all occupants are in their respective offices and 0 for all the other location vectors. Suppose that this posterior is used as the prior distribution for the next time period. If any of the occupants is moving during this new time period, we cannot update anymore since it will result in a posterior for this new time period in which all probabilities are 0.

This phenomenon is known as *Cromwell's rule*, see [8], which is best explained by the statement: *If the prior probability assigned to a hypothesis is 0 or 1, then, by Bayes' theorem, the posterior probability (probability of the hypothesis, given the evidence) is forced to be 0 or 1 as well; no evidence, no matter how strong, could have any influence.* In order to overcome this issue and still use the idea behind sequential Bayesian updating, we will use the following idea for algorithm 3:

Starting with an initial distribution $p^0_{loc}(s)$ we calculate the posterior distribution for the first time period. Next, we define the prior for the second time period as the average of the initial distribution and the posterior distribution of the first time period. We calculate the posterior for the second time period. The prior for the third time period is defined as the average of the initial distribution and the posteriors of the first two time periods. This process continues until we have the posterior of the last time period. The estimated stationary distribution is the time average of all posterior distributions.

This idea is translated into the following pseudo-code:

Algorithm 3:

Inference$(Z, P(Z_\tau|S_\tau, S_{\tau-1}))$
Initialization: $p^0_{loc}(s) = \frac{1}{|\mathcal{S}|}\ \forall s \in \mathcal{S}$
for $\tau = 1$ *to* T **do**
  $\hat{p}_{loc}(\tau, s) = P(S_\tau = s|Z_1 = z_1, ..., Z_\tau = z_\tau) = \frac{\sum_{s'} P(Z_\tau=z_\tau|S_\tau=s,S_{\tau-1}=s')p^{\tau-1}_{loc}(s)\hat{p}_{loc}(\tau-1,s')}{\sum_s \sum_{s'} P(Z_\tau=z_\tau|S_\tau=s,S_{\tau-1}=s')p^{\tau-1}_{loc}(s)\hat{p}_{loc}(\tau-1,s')}$
  $p^\tau_{loc}(s) = \frac{\tau}{\tau+1}p^{\tau-1}_{loc}(s) + \frac{1}{\tau+1}\hat{p}_{loc}(\tau, s)$
end
$\hat{\theta}_s = \frac{1}{T}\sum_{\tau=1}^{T} \hat{p}_{loc}(\tau, s)$

We also know that $\hat{p}_{loc}(0, s)$ is equal to 1 if s is such that all occupants are in their respective offices and 0 otherwise. This is a consequence of the decision to let all occupants start in their personal offices during the data generation ($\tau = 0$). A nice property of this updating rule is that the prior for a particular time period is influenced equally much by all the previous time periods. In table 7 we can see the summarized results of algorithm 3 for the three examples.

As can be seen from the results for algorithm 3, the results look very promising. For all three cases, we are able to estimate the fractions of time that the two occupants in each of the pairs are interacting with a margin of less then 0.5%. However, we still did not use all information that is available to define a suitable prior distribution for each of the time periods. In section 6.6 we will present an iterative algorithm that uses more information to define the prior distribution.

6.6 Algorithm 4

The main feature of this algorithm is that we use more information as compared to algorithm 3 to define the prior distribution at each of the time periods.

In section 6.5 we have seen that an informative prior distribution is essential for the inference to give useful results. By using the update rule as proposed in algorithm 3, we use the posterior distributions $P(S_\tau = s|Z_1 = z_1, ..., Z_\tau = z_\tau) \quad \tau = 1, ..., \tau^* - 1$, together with an initial distribution p^0_{loc}, to define the prior distribution at time period τ^*. However, we did not use the sensor signals $z_{\tau^*}, ..., z_T$ and the information that is captured in these sensor signals to define the prior distribution at time period τ^*. In order to use this extra information, we propose an iterative version of algorithm 2, which can be described by the following steps:

1. Using an initial distribution θ we will calculate the posterior distributions $P(S_\tau = s|Z_1 = z_1, ..., Z_\tau = z_\tau) \quad \tau = 1, ..., T$.
2. The prior distribution is changed into the average over all time periods of the posterior distributions.
3. Steps 1 and 2 are repeated until convergence of the prior distribution can be detected.

In pseudo-code, the algorithm is:

Algorithm 4:

Inference$(Z, P(Z_\tau|S_\tau, S_{\tau-1}), tol)$
Initialization: $\theta_s = \frac{1}{|\mathcal{S}|} \; \forall s \in \mathcal{S}$
for $\tau = 1$ *to* T **do**
  $\hat{p}_{loc}(\tau, s) = P(S_\tau = s|Z_1 = z_1, ..., Z_\tau = z_\tau) = \frac{\sum_{s'} P(Z_\tau = z_\tau|S_\tau = s, S_{\tau-1} = s')\theta_s \hat{p}_{loc}(\tau-1, s')}{\sum_s \sum_{s'} P(Z_\tau = z_\tau|S_\tau = s, S_{\tau-1} = s')\theta_s \hat{p}_{loc}(\tau-1, s')}$
end
$\hat{\theta}_s = \frac{1}{T} \sum_{\tau=1}^{T} \hat{p}_{loc}(\tau, s)$
while $||\theta - \hat{\theta}||_\infty > tol$ **do**
  $\theta = \hat{\theta}$
  repeat steps 3-6
end

We also know that $\hat{p}_{loc}(0, s)$ is equal to 1 if s is such that all occupants are in their respective offices and 0 otherwise. This is a consequence of the decision to let all occupants start in their personal offices during the data generation ($\tau = 0$). In table 7 we can see the summarized results of algorithm 4 for the three examples.

We can see that algorithm 4 gives a better result for the first case, a much better result for the second case and a slightly better result for the third case. However, this algorithm has one drawback. Like algorithm 3, this algorithms only uses the forward probabilities $P(S_\tau = s|Z_1 = z_1, ..., Z_\tau = z_\tau) \quad \tau = 1, ..., T$ to improve the stationary distribution. However, the backward probabilities $P(Z_{\tau+1} = z_{\tau+1}, ..., Z_T = z_T|S_\tau = s) \quad \tau = 1, ..., T - 1$ may contain information that can be used to give a better estimation of

the stationary probabilities. In section 6.7 we will present an algorithm that will use both forward and backward probabilities to improve the estimation.

6.7 Algorithm 5

The main feature of this algorithm is the introduction of backward probabilities to resolve ambiguity better. First we will see an example in which the introduction of backward probabilities is beneficial.

Suppose that we know that at a particular time period occupants 3 and 4 are in their respective offices, occupants 1 and 2 are in a meeting room together and occupants 5 and 6 are both in the other meeting room. Based on the sensor data at the beginning from the next time period, we can see that from each meeting room one occupant transfers to the other meeting room and occupants 3 and 4 stay in their respective offices. But since we do not have any prior knowledge with regard to the likelihood of any of these transfers, we need to assume that, within each meeting room, both occupants are equally likely to switch meeting rooms. As a result, we have that both situations, $\{\{3\},\{4\},\{1,5\},\{2,6\}\}$ and $\{\{3\},\{4\},\{1,6\},\{2,5\}\}$, are equally likely to happen, each having a different effect on the final result.

Let us now consider the sensor data from the subsequent time period (the time period after the one with the ambigious situation). Suppose we know (based on sensor data) that the two occupants in one of the meeting rooms stay where they are and the occupants of the other meeting room go to their respective personal offices. From the last data frame of the sample z_τ of this time period we see that one meeting room is occupied and the personal offices of occupamts 1, 3, 4 and 5 are occupied as well. We can then conclude that occupants 2 and 6 are in the other meeting room at this third time period we consider. But we can also conclude that they were interacting the previous time period. Hence we know that occupants 1 and 5 were interacting during the previous time period. This means that for the second time period we consider, we can exclude the situation $\{\{3\},\{4\},\{1,6\},\{2,5\}\}$.

The idea we discussed above is the reason for the introduction of backward probabilities. The following algorithm (that we give in pseudo-code) allows us to use to backward probabilities to improve the estimation of the stationary probabilities:

Algorithm 5:

Inference$(Z, P(Z_\tau|S_\tau, S_{\tau-1}), tol)$
Initialization: $\theta_s = \frac{1}{|\mathcal{S}|}\ \forall s \in \mathcal{S}$
for $\tau = 1$ *to* T **do**
  $\hat{p}_{loc}(\tau, s) = \frac{P(Z_1=z_1,\ldots,Z_\tau=z_\tau,S_\tau=s)}{\sum_s P(Z_1=z_1,\ldots,Z_\tau=z_\tau,S_\tau=s)} = \frac{\sum_{s'} P(Z_\tau=z_\tau|S_\tau=s,S_{\tau-1}=s')\theta_s \hat{p}_{loc}(\tau-1,s')}{\sum_s \sum_{s'} P(Z_\tau=z_\tau|S_\tau=s,S_{\tau-1}=s')\theta_s \hat{p}_{loc}(\tau-1,s')}$
end
for $\tau = T-1$ *to* 1 **do**
  $\tilde{p}_{loc}(\tau, s) = \frac{P(Z_{\tau+1}=z_{\tau+1},\ldots,Z_T=z_T|S_\tau=s)}{\sum_s P(Z_{\tau+1}=z_{\tau+1},\ldots,Z_T=z_T|S_\tau=s)} = \frac{\sum_{s'} P(Z_{\tau+1}=z_{\tau+1}|S_{\tau+1}=s',S_\tau=s)\theta_{s'} \tilde{p}_{loc}(\tau+1,s')}{\sum_s \sum_{s'} P(Z_{\tau+1}=z_{\tau+1}|S_{\tau+1}=s',S_\tau=s)\theta_{s'} \tilde{p}_{loc}(\tau+1,s')}$
end
$\hat{\theta}_s = \frac{1}{T}\sum_{\tau=1}^{T} P(S_\tau = s|Z_1 = z_1, ..., Z_T = z_T) = \frac{1}{T}\sum_{\tau=1}^{T} \frac{\hat{p}_{loc}(\tau,s)\tilde{p}_{loc}(\tau,s)}{\sum_s \hat{p}_{loc}(\tau,s)\tilde{p}_{loc}(\tau,s)}$
while $||\theta - \hat{\theta}||_\infty > tol$ **do**
  $\theta = \hat{\theta}$
  repeat steps 3-9
end

The normalization (division by the sum over all location vectors) of the forward and backward probabilities was implemented for numerical reasons. However, this will not influence the result as $\hat{p}_{loc}(\tau, s)$ and $\tilde{p}_{loc}(\tau, s)$ at every time period are divided by the same normalization constant for all states s. From the procedure with which the data was generated, we know that $\hat{p}_{loc}(0, s) = 1$ if s is such that all occupants are in their respective offices and 0 otherwise. We also have that $\forall s\ \tilde{p}_{loc}(T, s) = 1$ (since Z_T is the last sequence of sensor signals).

Before we look at the results, we will prove an important property of the algorithm. The update rule in line 9 of the algorithm can be written as $\hat{\theta}_s = \frac{1}{T}\sum_{\tau=1}^{T} P(S_\tau = s|Z_1 = z_1, ..., Z_T = z_T, \theta)$. So the new estimate for the stationary probabilities is the time-average of the posterior probabilities under the old estimate. If one would apply the concept of expectation maximization (see section 3.3) to the observed data $\mathbf{Z}$, the unobserved data $\mathbf{S}$ and the stationary probability distribution θ (see table 6), it would result in exactly the same update rule for the parameters in the vector θ. This property of the algorithm is captured in the following theorem:

Theorem 2: *The update rule for the stationary probabilities in algorithm 5 is exactly the update rule for the stationary probabilities if the concept of expectation maximization is applied to the observed data* $\mathbf{Z}$*, the unobserved data* $\mathbf{S}$ *and the stationary probability distribution* θ*.*

Proof. We will first apply the concept of expectation maximization to to the observed data $\mathbf{Z}$, the unobserved data $\mathbf{S}$ and the stationary probability distribution θ. Next we will show that the resulting update rule is exactly the update rule of algorithm 5.

Recall that $\mathbf{S} = (S_1, ..., S_T)$ are the random location vectors at time period $\tau = 1, ..., T$. Also recall that $\mathbf{s} = (s_1, ..., s_T)$ is a realization of $\mathbf{S}$. Moreover, recall that $\mathbf{Z} = (Z_1, ..., Z_T)$ are the sequences of sensor signals as defined in section 6.4 for time periods $\tau = 1, ..., T$. Lastly, recall that $\theta = (\theta_1, ..., \theta_{|\mathcal{S}|})$ is the stationary probability distribution over all possible location vectors of scheduled locations for all occupants.

Next, we define $L(\theta, \mathbf{s}, \mathbf{z})$=$P(\mathbf{S} = \mathbf{s}, \mathbf{Z} = \mathbf{z}|\theta)$. We have that this can be written as:

$$P(\mathbf{S} = \mathbf{s}, \mathbf{Z} = \mathbf{z}|\theta) = \mathbb{1}\{\mathbf{s} \to \mathbf{z}\} \prod_{\tau=1}^{T} \theta_{s_\tau} \tag{9}$$

Here $\mathbb{1}\{\mathbf{s} \to \mathbf{z}\} = 1$ denotes the fact that the sequence of location vectors $\mathbf{s}$ results in the sequences of sensor signals $\mathbf{z}$. We also need the conditional distribution $\mathbf{S}|\mathbf{Z} = \mathbf{z}, \theta^k$ which is the conditional distribution of the sequences of location vectors given the sequence of sensor signals and a current estimate of the distribution θ. Using Bayes' rule we find that the pdf of this conditional distribution is given by:

$$P(\mathbf{S} = \mathbf{s}|\mathbf{Z} = \mathbf{z}, \theta^k) = \frac{\mathbb{1}\{\mathbf{s} \to \mathbf{z}\} \prod_{\tau=1}^{T} \theta^k_{s_\tau}}{\sum_{\mathbf{s}'} \mathbb{1}\{\mathbf{s}' \to \mathbf{z}\} \prod_{\tau=1}^{T} \theta^k_{s'_\tau}} \tag{10}$$

Now we are able to find an expression for $E_{\mathbf{S}|\mathbf{Z},\theta^k}\ log(L(\theta, \mathbf{s}, \mathbf{z}))$:

$$E_{\mathbf{S}|\mathbf{Z},\theta^k}\ log(L(\theta, \mathbf{s}, \mathbf{z})) = \sum_{\mathbf{s}} \frac{\mathbb{1}\{\mathbf{s} \to \mathbf{z}\} \prod_{\tau=1}^{T} \theta^k_{s_\tau}}{\sum_{\mathbf{s}'} \mathbb{1}\{\mathbf{s}' \to \mathbf{z}\} \prod_{\tau=1}^{T} \theta^k_{s'_\tau}} \cdot \sum_{\tau=1}^{T} log(\theta_{s_\tau}) \tag{11}$$

Note that we do not need to consider the logarithm of $\mathbb{1}\{\mathbf{s} \to \mathbf{z}\}$ in calculating the log-likelihood since we only consider states $\mathbf{s}$ for which $\mathbb{1}\{\mathbf{s} \to \mathbf{z}\} = 1$ and hence $log(\mathbb{1}\{\mathbf{s} \to \mathbf{z}\}) = 0$. When we rewrite the expression in (11) such that we sum over all possible location vectors s instead over all possible sequences $\mathbf{s}$, we get the equivalent expression:

$$E_{\mathbf{S}|\mathbf{Z},\theta^k}\ log(L(\theta, \mathbf{s}, \mathbf{z})) = \sum_{j=1}^{|\mathcal{S}|} log(\theta_j) \cdot (\sum_{\tau=1}^{T} \frac{\sum_{\mathbf{s}} \mathbb{1}\{\mathbf{s} \to \mathbf{z}\}\mathbb{1}\{s_\tau = j\} \prod_{\tau=1}^{T} \theta^k_{s_\tau}}{\sum_{\mathbf{s}'} \mathbb{1}\{\mathbf{s}' \to \mathbf{z}\} \prod_{\tau=1}^{T} \theta^k_{s'_\tau}}) = \sum_{j=1}^{|\mathcal{S}|} log(\theta_j) \cdot (\sum_{\tau=1}^{T} \alpha_j^\tau) \tag{12}$$

Note that $\alpha_j^\tau = \frac{\sum_{\mathbf{s}} \mathbb{1}\{\mathbf{s} \to \mathbf{z}\}\mathbb{1}\{s_\tau = j\} \prod_{\tau=1}^{T} \theta^k_{s_\tau}}{\sum_{\mathbf{s}'} \mathbb{1}\{\mathbf{s}' \to \mathbf{z}\} \prod_{\tau=1}^{T} \theta^k_{s'_\tau}} = P(S_\tau = j|\mathbf{Z} = \mathbf{z}, \theta^k)$. Maximizing the expression in (12)

yields solving the following optimization problem:

$$\begin{aligned} \underset{\theta}{\text{maximize}} \quad & \sum_{j=1}^{|\mathcal{S}|} log(\theta_j) \cdot (\sum_{\tau=1}^{T} \alpha_j^\tau) \\ \text{subject to} \quad & \sum_{j=1}^{|\mathcal{S}|} \theta_j = 1. \end{aligned} \tag{13}$$

To solve this optimization problem, we will use the method of Lagrange multipliers. Given that our problem is a convex optimization problem, due to the concavity of our objective function and the linearity of the constraint, we know that we can find the optimal solution of our problem by solving:

$$\nabla f(\theta) = \lambda \nabla g(\theta) \tag{14}$$

$$f(\theta) = \sum_{j=1}^{|\mathcal{S}|} log(\theta_j) \cdot (\sum_{\tau=1}^{T} \alpha_j^\tau) \tag{15}$$

$$g(\theta) = \sum_{j=1}^{|\mathcal{S}|} \theta_j - 1 \tag{16}$$

$$\lambda \in \mathbb{R} \tag{17}$$

Determining the gradients of $f(\theta)$ and $g(\theta)$ and plugging these into equation (14) yields:

$$\begin{bmatrix} \frac{\sum_{\tau=1}^{T} \alpha_1^\tau}{\theta_1} \\ \vdots \\ \frac{\sum_{\tau=1}^{T} \alpha_{|\mathcal{S}|}^\tau}{\theta_{|\mathcal{S}|}} \end{bmatrix} = \lambda \begin{bmatrix} 1 \\ \vdots \\ 1 \end{bmatrix} \tag{18}$$

It can be easily seen that the solution to equation (18) is given by:

$$\theta_j = \frac{\sum_{\tau=1}^{T} \alpha_j^\tau}{\lambda} \;\; j = 1, ..., |\mathcal{S}| \tag{19}$$

In order to ensure primary feasibility, that is fulfilling the equality constraint, we set $\lambda = \sum_{j=1}^{|\mathcal{S}|} \sum_{\tau=1}^{T} \alpha_j^\tau$. In this way, we ensure that $\sum_{j=1}^{|\mathcal{S}|} \theta_j = 1$. Now recall that $\alpha_j^\tau = P(S_\tau = j | \mathbf{Z} = \mathbf{z}, \theta^k)$. Hence we can write:

$$\lambda = \sum_{j=1}^{|\mathcal{S}|} \sum_{\tau=1}^{T} \alpha_j^\tau = \sum_{\tau=1}^{T} \sum_{j=1}^{|\mathcal{S}|} \alpha_j^\tau = \sum_{\tau=1}^{T} \sum_{j=1}^{|\mathcal{S}|} P(S_\tau = j | \mathbf{Z} = \mathbf{z}, \theta^k) = \sum_{\tau=1}^{T} 1 = T \tag{20}$$

This means that the optimal updating rule for θ is:

$$\theta_j = \frac{\sum_{\tau=1}^{T} \alpha_j^\tau}{T} = \frac{\sum_{\tau=1}^{T} P(S_\tau = j | \mathbf{Z} = \mathbf{z}, \theta^k)}{T} \;\; j = 1, ..., |\mathcal{S}| \tag{21}$$

Since this is exactly the updating rule of algorithm 5, we have proven that this algorithm is an example of expectation maximization. □

In table 7 we can see the summarized results for algorithm 5 for the three examples. Using algorithm 5 we can estimate all the interaction probabilities with an absolute error of at most 0.0013. We also see that the introduction of the backward probabilities has a positive effect on the results for each of the three examples.

6.8 Summary of the results

In table 7 we see the summarized results for each of the algorithms that is applied to each of the three examples (see chapter 5).

The results for the last algorithm are very promising. In chapter 7 we will investigate whether this algorithm can also give such good results if it is applied to a larger example. We will first discuss the main issue for the inference: scalability.

		Algorithm 1	Algorithm 2	Algorithm 3	Algorithm 4	Algorithm 5
Example 1	AAE	0.0319	0.0018	0.0004	0.0002	0.0002
	MAE	0.0476	0.0029	0.0014	0.0009	0.0009
Example 2	AAE	0.1100	0.0231	0.0015	0.00004	0.000008
	MAE	0.1546	0.0342	0.0047	0.0002	0.00004
Example 3	AAE	0.0149	0.0030	0.0015	0.0014	0.0003
	MAE	0.0236	0.0064	0.0046	0.0046	0.0013

Table 7: Absolute average error (AAE) and maximal absolute error (MAE) for each of the algorithms and all examples

6.9 Scalability

In section 4.5 we discussed the issue of scalability for the model to generate sensor data. By considering the worst-scenario, we argued that the number of maximal independent sets can be as large as the number of partitions of the set of occupants (denoted by B_N). This was explained by the fact that, in the worst-case scenario, the interaction graph is completely connected and the set of meetings is as large as possible, namely the number of non-empty subsets of the set of occupants.

Due to the lack of prior knowledge, we need to assume for the inference that the worst-case scenario is a possible scenario. Therefore we have to assume that the number of maximal independent sets that can be sampled is as large as the number B_N. If the location allocation procedure is not randomized, we know that the number of possible location vectors (one location for each occupant) is equal to the number B_N. However, we have that for the inference the best-case scenario, under the assumption that we have no prior knowledge, would be the situation in which the number of possible location vectors is equal to the number B_N.

If the location allocation is a randomized procedure (as we have seen in section 5.5) we have that the number of possible location vectors is even bigger than B_N. This is explained by the fact that there are maximal independent sets that correspond to more than one location vector. If we look at the numbers in section 4.5, we see that scalability is thus an issue for the inference as well. This holds especially if the location allocation procedure is randomized.

One remark with regard to the large number of location vectors must be made. If we use algorithm 5, which is an iterative algorithm, we see that after the first iteration many location vectors will have probability 0 of occurring. Starting with 405 possible locations vectors for example 1, we see that after one iteration of algorithm 5, we only have 63 location vectors that have a non-zero probability of occurring. This is explained by the fact that in the first iteration many location vectors can be excluded based on the sensor data alone. However, scalability remains an issue for the inference of the social network.

7 Application of the model to generate data and inference of the social network for a larger example

In this chapter we will apply the model to generate data and the last inference algorithm to a larger example. In this way, we will be able to have a better understanding of the performance of the algorithm. Moreover, we can gain insight in the applicability of the algorithm to real office buildings, since these will have a higher number of occupants. We will make use of the elaborate procedure to determine the stationary probability distribution and explore how it can be used to generate more realistic data.

The example that we will consider has twelve occupants (such that we have $\frac{12 \cdot 11}{2} = 66$ pairs) who all have a personal office. Morover there will be four meeting rooms. The case can be summarized by figure 14 and table 8.

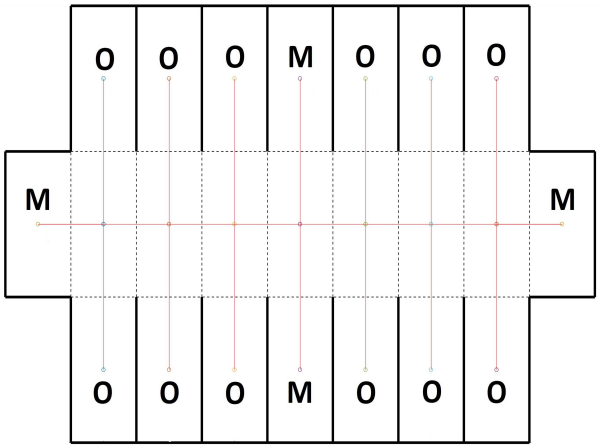

Figure 14: The office building, where the twelve personal offices are marked with the symbol **O**, the four meeting rooms are marked with the symbol **M** and the other cells form the paths between the different work areas. The bold black lines are the walls of the building.

Occupant	1	2	3	4	5	6	7	8	9	10	11	12
1	0	0.10	0.10	0.05	0	0	0.05	0	0	0.05	0	0
2	0.10	0	0.10	0	0.05	0	0	0.05	0	0	0.05	0
3	0.10	0.10	0	0	0	0.10	0	0	0.10	0	0	0.10
4	0.05	0	0	0	0.10	0.10	0.05	0	0	0.05	0	0
5	0	0.05	0	0.10	0	0.10	0	0.05	0	0	0.05	0
6	0	0	0.10	0.10	0.10	0	0	0	0.10	0	0	0.10
7	0.05	0	0	0.05	0	0	0	0.10	0.10	0.05	0	0
8	0	0.05	0	0	0.05	0	0.10	0	0.10	0	0.05	0
9	0	0	0.10	0	0	0.10	0.10	0.10	0	0	0	0.10
10	0.05	0	0	0.05	0	0	0.05	0	0	0	0.10	0.10
11	0	0.05	0	0	0.05	0	0	0.05	0	0.10	0	0.10
12	0	0	0.10	0	0	0.10	0	0	0.10	0.10	0.10	0

Table 8: Interaction matrix for the example of an office building with twelve occupants

The structure of the interaction matrix is as follows. For the groups $\{1, 2, 3\}$, $\{4, 5, 6\}$, $\{7, 8, 9\}$, $\{10, 11, 12\}$ and $\{3, 6, 9, 12\}$ we have that, within each group, every occupant meets all other occupants with probability 0.10. For the groups $\{1, 4, 7, 10\}$,and $\{2, 5, 8, 11\}$ we have that, within each group, every occupant meets all other occupants with probability 0.05.

Next, we determine which meetings can happen in the office building. Besides a meeting for each individual occupant, we have that there is a meeting for each pair of occupants that is connected in the interaction graph. Moreover, there is a meeting for every maximal connected group of occupants that contains more than two occupants. For this case we have that these groups are: $\{1,2,3\}$,$\{4,5,6\}$, $\{7,8,9\}$, $\{10,11,12\}$, $\{1,4,7,10\}$, $\{2,5,8,11\}$ and $\{3,6,9,12\}$. Finding all maximal independent sets in the graph of all these meetings yields $L = 8249$ maximal independent sets.

In order to find an approriate stationary distribution π, we will wil solve the following optimization problem:

$$\begin{aligned} \underset{\pi}{\text{minimize}} \quad & \sum_{l=1}^{8249} c_l^2 \pi_l \\ \text{subject to} \quad & A\pi = p \\ & \tilde{A}\pi = \frac{p}{5} \\ & \pi_l \geq 0,\ l = 1, \ldots, 8249. \\ & \sum_{l=1}^{8249} \pi_l = 1. \end{aligned}$$

Here $A, \tilde{A} \in \{0,1\}^{66 \times 8249}$ and $p \in [0,1]^{66}$. The coefficients c_l represent the number of meetings (with more than one occupant) that take place if the maximal independent set M_l is chosen. The choice of the objective function and the addition of the constraint were implemented to ensure that π reflects certain properties that may occur in real office buildings. In this way, the sensor data will be more realistic.

By using the objective function as defined, we have that maximal independent sets with many meetings will occur as little as possible. In this way, we will see rather often that a maximal independent set is chosen with a small number of meetings. This was done to ensure that the distribution π resembles the fact that meetings occur spontaneously rather than simultaneously, which may be the case in reality.

The matrix $\tilde{A}$ is defined such that $\tilde{a}_{r,l} = \mathbb{1}\{\text{occupants in } r \text{ meet in larger meeting in } M_l\} \cdot a_{r,l}$. Recall that a larger meeting was defined as a meeting with more than two occupants. By adding the constraint $\tilde{A}\pi = \frac{p}{5}$, we simulate the fact that 20% of the interaction within each pair happens during larger meetings and that 80 % of the interaction within each pair happens in an one-on-one meeting. For our case, this problem has a feasible solution. In case that there is no feasible solution, one can alter the constraints or the objective function such that a feasible solution exists while ensuring that π has certain properties.

With regard to the allocation of rooms to meetings, the following allocation rules are used:

- All individual meetings take place in the office of the respective occupants.
- All meetings with two occupants take place in the office of the occupant with the highest number (as can be found in table 8).
- All meetings with more than two occupants will take place in a meeting room. To allocate each of these meetings to a meeting room, we use the following procedure:
  - For each meeting that will take place in a meeting room, we will determine the participant with the highest number (such that each meeting gets a unique number).
  - The meetings will then be ordered based on these numbers. We also have ordered the meeting rooms (clockwise and starting with the meeting room on the top of the middle column). Each meeting will be assigned the corresponding meeting room.

This case has been explored for 1200, 2400, 4800 and 9600 time periods. We chose to explore different sizes of the data set to see to what extend the results can be explained by the fact that the data set has a fixed size. The results can be found in table 9.

	average absolute error	maximal absolute error
T=1200 (3 months)	0.0002	0.0033
T=2400 (6 months)	0.00006	0.0015
T=4800 (1 year)	0.000009	0.0001
T=9600 (2 years)	0.000009	0.0002

Table 9: Summary of the results for the example with twelve occupants

As we can see in table 9, the results of this algoritm are very promising. Based on a year of sensor data, we can estimate all interaction probabilities with an absolute error of at most 0.0001. We also see that for this example that doubling the amount of data does not improve the results, whereas taking away 50% or 75 % of the data does influence the results rather much.

8 Summary, discussion, ideas and recommandations

In this thesis we have investigated to what extend it is possible to infer a social network based on sensor data from lighting systems in an office building. We first summarize and discuss the contents of the previous chapters. Next, we discuss two ideas for the model to generate sensor data and an idea for the inference. We conclude this chapter with formulating several recommandations for further research.

8.1 Summary and discussion

We first developed a model to generate sensor data of an office building. We studied literature on the simulation of occupancy in office buildings. We found that each of the different approaches lacks the explicit modeling of interaction that we need in our model. Therefore, we proposed our own model to generate sensor data that enables us to incorporate the interaction explicitely.

Model to generate sensor data: the framework and the movement mechanism
The office building itself was modeled by a grid where each cell is a either a work area or a segment in the different paths. The movement of the occupants was modeled using a similar simple approach. Based on the interaction model, each occupant is assigned a work area at every time period. At the beginning of each time period, each occupants moves from the current location to the scheduled work area by walking along the grid cells at constant speed until the destination is reached.

Model to generate sensor data: modeling interaction amongst the occupants
The basis of our interaction model is an affiliation graph in which an occupant is connected to a meeting if the occupant is part of the meeting. By projecting this graph onto the set of meetings, we had a graph in which two meetings are connected if and only if they have an occupant in common, meaning that they cannot be scheduled at the same time. By finding the maximal indepent sets of this graph, we were able to group all possible meeting such that within each group of meetings, every occupant gets assigned to exactly one meeting.

A meeting schedule was generated by selecting one of the maximal independent sets at every time period according to a stationary probability distribution. Desired interaction between the occupants was incorporated in this probability distribution by selecting the probability distribution according to an appropriately defined minimization problem. We also discussed the fact that scalability is an issue for this model. We stated that, in the worst-case scenario, the number of maximal independent sets is equal to the number of partitions of the set of occupants. This creates a barrier for the application of the model to larger examples.

Model to generate sensor data: detailed location data and sensor data
Through our movement model and the interaction model, we can simulate detailed location data (the location of every occupant at each of the time steps). By implementing a simple sensor, we could translate this detailed location data into sensor data telling what grid cells are occupied at each of the time steps.

Inference of the social network
We presented three examples of an office building with six occupants and two meetings rooms. These three examples were used to evaluate the performance of the inference algorithms. We discussed the main challenge for the inference: the concept of ambiguity. In the transformation of location data into sensor data, information is lost. As a consequence, different location vectors result in the same sensor data. The phenomenon in which several states (location vectors) become indistinguishable was referred to as ambiguity.

Next, we proposed a series of Bayesian inference algorithms. Each algorithm made better use of the data and its structure. The first algorithm we proposed showed that ambiguity is of significant influence on the results if it is not resolved. The second algorithm we proposed was designed to partially resolve ambiguity. The third, fourth and fifth algorithm we proposed each further improved the results. We discussed the concept of expectation maximization. We also showed that the update rule of the fifth

algorithm is exactly the same as the update rule we would get if we would apply the concept of expectation maximization to our problem. For the three examples with six occupants, we were able to estimate all interaction probabilities with an absolute error of at most 0.0013.

We also discussed that the scalability issue of the model is an issue for the inference as well. For the model we argued that in the worst-case scenario, the number of maximal independent sets is equal to the number of partitions of the set of occupants. For the inference we have that the number of possible location vectors (the number of states we need to consider) is equal to the number of partitions of the set of occupants in the best-case scenario. If the location allocation procedure is randomized, the number of possible location vectors would be even larger. The fact that this issue plays a role in the inference is explained by the fact that we need to consider all possibilities due to a lack of prior knowledge.

Application of the model to generate sensor data and inference of the social network for a larger example
The model to generate sensor data and the last algorithm were also applied to an example with twelve occupants and four meeting rooms. This example was designed to be a realistic representation of a small office building. Based on a year of sensor data, the algorithm was able to estimate the interaction probabilities with an absolute error of at most 0.0001. Given that for this example the interaction probabilities were all either five or ten percent, we may conclude that this algorithm was able to estimate all the interaction probabilities very accurately.

8.2 Ideas for the model to generate sensor data and the inference

The model to generate sensor data
The model we proposed to generate sensor data is a closed system, meaning that the occupants never leave the building and that there are no visitors. If occupants would have the possibilty to leave the building and it would be possible for visitors to enter the building (and meet with occupants) the model would be a much better representation of reality.

In our model, we build a meeting schedule by choosing a maximal independent set from a stationary probability distribution at every time period. If the sequence of maximal independent sets would come from a Markov chain, it would allow us the choose shorter time periods (five minutes instead of half an hour). The main advantage of this would be that the dynamics of the movement would be more realistic as meetings would not all start at (approximately) the same time and the length of the meetings can be made variable. However, it remains important one can still control the interactions amongst occupants.

The inference
In this thesis, we attempt to reconstruct the entire social network explicitly, i.e. estimate the interaction probabilities for all pairs of occupants. However, one could also look for less explicit characteristics of the network. One example of such a characteristic is clustering within the social network. Another less specific characteristic of the network one may be interested in is interaction within a particular group of occupants, e.g. the managers of a company. Looking for less explicit characteristics may also partially solve the scalability issue we encountered with the inference in this project. Depending on the goal of the inference, it may be possible to aggregate states (location vectors) without compromising the performance of the inference.

8.3 Recommandations

Based on the contents of this thesis, we formulate the following recommandations for further research:

- Conduct further research with regard to the scalability of both the model and the inference since this is a barrier for applying the model and inference to real office buildings.
- Conduct further research with regard to the ideas for the model and the inference. Especially investing the ideas for the model can contribute to formulating a more realistic model.
- Conduct research with regard to incorporating prior information to reduce ambiguity. An example of prior information that could be used is a calendar of meetings that are going to happen.

- Develop a test setting in an office building. This could be done by equipping (part of) a floor of an office building with sensors. In this way, it can be analyzed which characteristics of the model fit reality and, more importantly, which characteristics of the model do not fit reality.
- Investigate if the concepts discussed in this thesis can be applied to similar settings in which there is both observable and unobservable data. One example of such a setting is: inference of the location of ships based on radar data.

References

[1] Bron, C. & Kerbosch, J. (1973). *Algorithm 457: finding all cliques of an undirected graph.* Communications of the ACM, vol 16 (9), pp 575-577

[2] Carson, C., Belongie, S., Greenspan, H., & Malik, J. (2002). *Blobworld: image segmentation using expectation-maximization and its application to image querying.* IEEE Transactions on Pattern Analysis and Machine Intelligence, vol 24 (8), pp 1026-1038

[3] Cazals, F. & Karande, C. (2008). *A note on the problem of reporting maximal cliques.* Theoretical Computer Science, vol 407 (1), pp 564-568

[4] Dempster, A.P., Laird, N.M. & Rubin, D.B. (1977). *Maximum likelihood for incomplete data via the EM algorithm.* Journal of the Royal Statistical Society, Series B. vol 39 (1), pp 1-38

[5] Easly, D. & Kleinberg, J. (2010). *Networks, Crowds and Markets.* Cambridge University Press

[6] Feng, X., Yan, D. & Hong, T. (2015). *Simulation of occupancy in buildings.* Energy and buildings, vol 87 (1), pp 348-359

[7] Johnson, D.S. & Yannakakis, M. (1988). *On generating all maximal independent sets.* Information Processing Letters, vol 26 (3), pp 119-123

[8] Lindley, D. (1991). *Making decisions* (2 ed.). Wiley. pp 104. ISBN 0-471-90808-8.

[9] Page, J., Robinson, D., Morel, N.& Scartezzini, J.-L. (2008). *A generalised stochastic model for the simulation of occupant presence.* Energy and buildings, vol 40 (2), pp 83-98

[10] Rabiner, L.R. (1989). *A tutorial on hidden Markov models and selected applications in speech recognition.* Proceedings of the IEEE vol 77 (2), pp 257-286

[11] Tomita, E., Tanaka, A. & Takahashi, H. (2006) *The worst-case time complexity for generating all maximal cliques and computational experiments.* Theoretical Computer Science, vol 363 (1), pp 28-42

[12] Tsukiyama, S., Ide, M., Ariyoshi, I., & Shirakawa, I. (1977) *A new algorithm for generating all the maximal independent sets.* SIAM Journal on Computing, vol 6 (3), pp 505-517

[13] Wang, D., Federspiel, C.C. & Rubinstein, F. (2005). *Modeling occupancy in single person offices.* Energy and buildings, vol 37 (2), pp 121-126

[14] Wang, D., Yan, D. & Jang, Y. (2011). *A novel approach for building occupancy simulation.* Building Simulation, vol 4 (2), pp 149-167

[15] Yu, T. (2015). *Modeling occupancy behavior for energy efficiency and occupants comfort management in intelligent buildings.* IEEE Ninth International Conference on Machine Learning and Applications (ICMLA), pp 726-731

Appendix

In table 10 the results can be found for algorithm 1 that is applied to example 1. The middle column displays the realized fractions of interaction. The difference between these fractions and the interaction probabilities in the interaction graph (see figure 10) are explained by the fact that we have a finite sample of maximal independent sets (T=4800). The rightmost column displays the estimated probabilities of interaction that were found using the estimated probability distribution of locations. This probability distribution was found using algorithm 1 (see 6.3).

Pair of occupants (i, j)	True fractions of interaction (ρ_{ij})	Estimated probabilities of interaction $(\hat{\rho}_{ij})$
1,2	0.1044	0.0568
1,3	0.1048	0.0679
1,4	0	0.0322
1,5	0	0.0276
1,6	0	0.0263
2,3	0.1038	0.0671
2,4	0	0.0325
2,5	0	0.0265
2,6	0	0.0256
3,4	0.0496	0.0581
3,5	0	0.0317
3,6	0	0.0298
4,5	0.1023	0.0655
4,6	0.0990	0.0652
5,6	0.1046	0.0586

Table 10: Results for algorithm 1 and example 1

We see that the interaction of the pairs that do not interact frequently ($\rho_{ij} < 0.05$) are overestimated whereas the interactions of the pairs that do interact frequently ($\rho_{ij} > 0.05$) are underestimated. This is explained by the fact that algorithm does not resolve ambiguity at all as has been argued in section 6.3.