UNIVERSITY OF THE WITWATERSRAND

School of Computer Science and Applied Mathematics
Faculty of Science
HONOURS RESEARCH REPORT

Automatic Lecture, Tutorial, and Laboratory University Venue Allocation for Varying Class Sizes

Uvir Bhagirathi

Supervisor(s): Ritesh Ajoodha

November 2018, Johannesburg

Declaration
University of the Witwatersrand, Johannesburg
School of _
SENATE PLAGIARISM POLICY

I, _ , (Student number: _ _ _ _ _ _ _ _ _ _ _ _ _)
am a student registered for _ in the year _ _ _ _ _ _ .

I hereby declare the following:

- I am aware that plagiarism (the use of someone else's work without their permission and/or without acknowledging the original source) is wrong.
- I confirm that ALL the work submitted for assessment for the above course is my own unaided work except where I have explicitly indicated otherwise.
- I have followed the required conventions in referencing the thoughts and ideas of others.
- I understand that the University of the Witwatersrand may take disciplinary action against me if there is a belief that this in not my own unaided work or that I have failed to acknowledge the source of the ideas or words in my writing.

Signature: _ _ _ _ _ _ _ _ _ _ _ _ _ **Date**: _ _ _ _ _ _ _ _ _ _ _ _ _

Automatic Lecture, Tutorial, and Laboratory University Venue Allocation for Varying Class Sizes

Uvir Bhagirathi
Department of Computer Science and Applied Mathematics
University of Witwatersrand, Johannesburg
South Africa
uvir127@gmail.com

Ritesh Ajoodha
Department of Computer Science and Applied Mathematics
University of Witwatersrand, Johannesburg
South Africa
ritesh.ajoodha@wits.ac.za

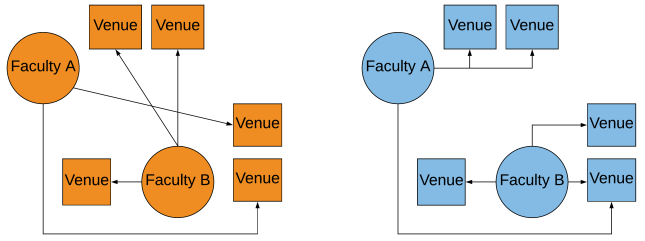
***Abstract*—The following paper addresses an approach to the problem of venue allocation. Currently, classes are assigned to venues that are inadequate in terms of size and are inconveniently situated from the respective faculty building of the class. This results in classes from different schools being allocated venues across the university. Therefore, the aim of this research is to design a program/algorithm to aid appropriate and effective class allocation. In doing so, a Bayesian network was implemented to generate synthetic data and Bayesian estimation was used to learn the parameters of the network. Thereafter, the network was scored using a *greedy* hill-climbing approach with random restart to ensure a global optimal solution. The program was tested with various data-sets of different sizes, displaying the scalability of the program. This approach was more efficient than a brute force solution to the problem as the system produced the best allocations of venues to classes in time complexity of $O(kn^2)$. It also favours closer venues over venues with slightly better accommodation. Ultimately, this solution will allow universities and other academic institutions to allocate the best venue, in terms of size and distance, to each class, improving the overall functioning of the university or academic institution.**

***Index Terms*—Scoring function, Heuristic search, Allocation, Bayesian network, Greedy Hill-Climbing**

I. Introduction

Universities attract large amounts of students for which each faculty has to facilitate. Within each faculty, students are registered for various courses under different schools and require appropriate learning environments for each course. While these universities do offer such environments there is often difficulty in allocating classes to these venues based on the needs required by different schools. The goal of this research project is to construct a system that allocates venues to classes of various sizes by taking into account the distance between the respective faculty buildings and the venues being used, the capacity of the venue as well as the number of students in the course.

In the current system of class-venue allocation, classes are often assigned to venues that are too small or big and far from the courses' faculty building. This results in classes being allocated to venues that are far away from each other. The issue comes about when classes from each faculty is spread across different venues throughout the entire campus (e.g. courses from humanities get allocated venues near the science faculty) when there are suitable venues close to the respective faculty building. This type of allocation leads to an unnecessary waste of time as students and staff have to walk to these distant venues. Figure 1 illustrates how the current allocation system assigns venues to classes versus the proposed systems way of allocating venues.



(a) Current Allocation System (b) Proposed Solution

Fig. 1: A graphical representation of how the current allocation system works versus how the proposed system works

In solving this problem, a Bayesian network (BN) model, an example of probabilistic models, was used to generate data of classes and venues. Probabilistic models utilize probability distributions as well as random variables in order to model an event. These models can, however, express uncertainty by modelling influence between variables. Variables of this network include attributes of the class and venue (size and type) as well as the distance to the respective faculty.

After the data was generated, a *greedy* hill-climbing algorithm was used to find an optimal network. A network of randomly allocated classes and venues is first created. A scoring function was then used to evaluate the effectiveness of the network model. The score of an allocation corresponds to how well the venue can accommodate the class assigned to it. Afterwards, a number of different operators are applied to the network. The algorithm stops once the score of the network does not better the current best score that it has found. This produces an optimal network.

The program developed in this research aims to allocate suitable venues to classes taking into consideration the distance

between the venue and the respective class faculty location as well as the size of the venue and the number of students in the class. The system is expected to produce a list of these allocations in at least $O(kn^2)$ time with each venue being best suited for the class that is assigned to it. This system will allow universities and other academic institutions to allocate the best venue to each class in a short amount of time, resulting in better functioning universities and institutions. Overall, this program is the stepping stone in developing a more robust scheduling system in universities and other academia.

The following sections of the research will address the background of the research, related work, methodology and results as well as future improvements.

II. Background

In this section BNs, Bayesian Estimation and the *Greedy* hill-climbing algorithm will be discussed. Probabilistic graphical models (PGMs), such as BNs, are used in various real world applications. *greedy* hill-climbing is a heuristic search algorithm that is used to traverse through the many combinations of allocations in the network. It produces a solution that is not necessarily the optimal solution.

Currently, class timetables are created highly inefficiently. This leads to many problems including overcrowding in classrooms, inadequate equipment to cater for lectures within these venues, overbooking of venues and venues being too distant between consecutive lectures. Teachers have expressed that effective teaching is not possible in overcrowded classrooms [Khan and Iqbal, 2012]. Allocating venues far apart from each other can cause an inconvenience as students will feel the need to rush out of the current class they are in to make it in time for the next class. This can also lead to students arriving late to class and disrupting lectures.

The program that has been developed in this research automates this system. The automated system allocates each course to the most suited venue according to a predetermined scoring function. This system minimizes the amount of extra seats in each venue and the distance between the venue and the respective faculty building of the class assigned to it. This allows for students and staff members to be more efficient when attending lectures as all venues will be close by.

The following subsections address BNs, Bayesian estimation as well as scoring functions.

A. Bayesian Network

A BN is a probabilistic graphical model in the form of a Directed Acyclic Graph (DAG) where each node is a variable and each edge is the conditional dependence between two variables [Koller et al., 2009]. BNs map the relationship between events in terms of probability. It shows how the occurrence of certain events influence the probability of other events occurring. A BN is used to generate the data. Figure 2 is an example of a BN [Pearl, 2014]:

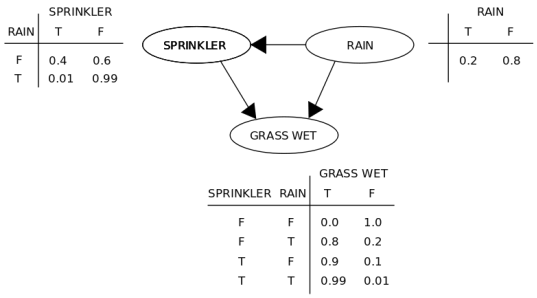


Fig. 2: Sprinkler Problem

It can be seen in Figure 2, each node has a probability of its occurrence and probability of its occurrence depending on the occurrence of other nodes. This kind of diagram is very useful when we have inter-dependent nodes and want to model outcomes and make decisions. In order to make these decisions, the probability of each nodes' occurrence needs to be known. These probabilities can be estimated using Bayesian estimation.

B. Bayesian Estimation

Bayesian estimation (BE) is a parameter learning method which is fairly similar to Maximum Likelihood Estimation (MLE), a commonly used learning tool for observable data [Ajoodha and Rosman, 2018]. BE views any event that has uncertainty as a random variable with a distribution over it [Ajoodha and Rosman, 2018]. BE uses Bayes' Theorem (Figure 3) and other functions to calculate the parameter values.

$$P(A|B) = \frac{P(B|A)P(A)}{P(B)}$$

Fig. 3: Bayes' Theorem

Bayesian analysis is a convenient setting for many models including hierarchical models. However, it can sometimes get computationally intensive when models involve many variables. Once the parameters of the model have been estimated, an assessment of the model has to be conducted. This can be done using a scoring function.

C. Scoring Function

A scoring function is used to evaluate the appropriateness of a network. Depending on the rules of the scoring function, each network is allocated a score which shows how well the model fits the observed training data [Ajoodha and Rosman, 2018].

There are many scoring functions however, the preferred method that is used is the *greedy* hill-climbing algorithm.

1) Greedy Hill-Climbing: The *greedy* hill-climbing algorithm is used in optimization problems where we need to minimize or maximize a function [Tsamardinos et al., 2006]. It is a heuristic search algorithm, which means that it might not find the optimum solution. However, it will provide a local minimum or maximum in reasonable time. The greedy part of the algorithm implies that it moves in which ever direction optimizes the function. The problem with this algorithm is that it may not always output the optimum network structure but this can be solved using backtracking or random restarts.

III. Related Work

Score-based structure learning defines the basis to evaluate how well a BN fits the data, then traverses over the network for a more optimal network which produces a better score. A score-based function is seen as an optimization problem where the produced result is not necessarily the optimal. However, the search space is super-exponential in size so we resort to heuristic search techniques.

A score-based structure is chosen as it allows for viable computations as it preserves basic score properties and indicates the effect of changes in the network structure. In this section we will discuss the *greedy* hill-climbing heuristic search algorithm and the scoring function that is used to score the BN in this research.

A. Heuristic Search

A heuristic search algorithm is a method that finds a sub-optimal solution in reasonable time. It increases efficiency by sacrificing completeness. A classic example is the travelling salesman problem. [Christofides, 1976]

Greedy hill-climbing is an example of a heuristic search used in optimisation problems. In an optimization problem, we pursue some optimum solution to a network model. Any given state of a network can be a solution. Solutions can be evaluated for it to be comparable. In *greedy* hill-climbing, an initial solution is then expanded into multiple neighboring solutions. The best network from those solutions is selected. This process continues until there are no better neighboring solutions. The final solution can be a sub-optimum solution as it is based on the generation of the neighboring networks and the scoring function used to evaluate those networks.

B. Score-Based Learning

When a network is generated there is no way of knowing its worth until it is scored. A scoring function is used to do this. The *greedy* hill-climbing algorithm is heavily dependent on the scoring of each network to produce a suitable solution. The score of the network indicates how well the network is suited for the task required. [Munteanu and Cau, 2000]

IV. Methodology

The following section addresses the methods utilized in solving the venue allocation problem. The structure of the BN utilized is also discussed as well as the way in which each node in the model interacts with one another. Additionally, a description of the data in the network will be given.

A. Motivation

The problem of venue allocation is difficult to solve as checking the compatibility of every class with every venue is an NP-Hard problem. For example, assume that there is a class c that has to be allocated to one of the n number of venues available. The class is firstly allocated to this first venue and then checked to see if it meets the necessary requirements. It is then allocated to the second venue and checked, then the third and so on until it is checked against the n^{th} venue. For a single c this can lead to a complexity of $O(n!)$ which is problematic for a large number of venues and classes where the complexity can be $O(c!n!)$.

Despite this, there are other downfalls to this method. The output of this method does guarantee an optimal solution to this problem; however, it is an NP-hard problem that will take a long time to run. The attempt to attain an adequate venue on short notice is then very time consuming especially for staff members and students, thus affecting the overall ability of the university to function productively and efficiently. The goal of this research project is to then find a more reasonable approach to solve this problem.

In my research I propose to produce a more feasible approach to the venue allocation problem by using a heuristic search to find a solution. This method can lower the complexity of the problem to at least $O(kn^2)$ which is a major improvement to the original complexity of $O(c!n!)$.

B. Network structure

The program allocates an adequate venue for each lecture depending on the following variables: the size of the class and venue, the type of class and venue and the distance of the venue from the courses' corresponding faculty. This is illustrated in Figure 4 where it can be seen that the venue chosen is dependent on the aforementioned variables. The distance between the class and venue, the size difference between the class and venue as well as the type of them, all influence the allocation score between that venue and class.

There is also a hidden variable, projector, which states if a venue or class has or requires a projector. This variable was added to influence better venue-class allocations. This is the only hidden variable in the program. It was used to test how easily the program can be influence to allocate venues to classes that require a projector. In the future other equipment and facilities can be added to the program to make sure the correct kind of venues are allocated to classes.

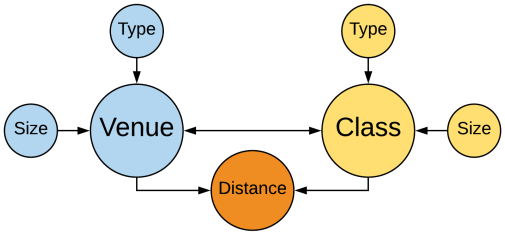


Fig. 4: Representation of the allocation algorithm

C. Data

The variables of the data used to train this network are represented in the Table I.

From Table I it can be seen that both the class and venue variables have a size and type attribute as well as a distance matrix. These attributes will be used to evaluate the probability of a venue being used for a class. The *course code* and *venue* variables are a form of meta-data that allows distinguish-ability between different classes and venues respectively.

D. Scoring

The *greedy* hill-climbing algorithm is a heuristic search algorithm which is used to find a local minimum solution to the problem. Since the algorithm does not produce the optimum solution all the time, random restarts are used to find several solutions, then the best of those solutions is chosen as the final solution.

The *greedy* hill-climbing algorithm starts with any random network. From this network, it iteratively tries to improve the network by performing different search operators. *Greedy* hill-climbing always chooses the network which provides the best score until the score cannot be improved any more. Due to the structure of the network, four search operators are used to find a solution to this problem. These search operators are discussed later on.

E. Scoring Function

The actual scoring of the class to venue allocation will depend on the following criteria:

- The size of the class and venue
- The type of class and venue
- The attributes of the class and venue
- The distance of the venue from the faculty of the class

This allows the score to be decomposable resulting in a more efficient algorithm [Heckerman et al., 1995]. The scoring function treats each allocation as a minimization problem. A reasonable positive score is given unless any of the following conditions are met. If a class is allocated to a venue that does not have enough seats, then a high positive score is given as the class cannot take place in that venue. However, if the size of the venue allocated surpasses the size of the class, then a low positive score is given with respect to the ratio between the class and venue size as the class can continue but there is waste of space that can be utilized by other classes.

A low positive score is given for any class-to-venue allocation where the types do not match. This is because certain classes may require specific facilities such as computers or work stations and therefore, cannot be allocated to a venue that does not accommodate these features. If the distance of a venue to the faculty of an allocated class is further than a certain threshold, then a low positive score is given. This will ensure that the closest venue is selected when possible. Lastly, if a venue clash occurs then a high positive score is given to that allocation, allowing for more clash-free allocations in the future. The pseudocode in Figure 5 illustrates these scoring features.

The best score of an allocation can be 0. This means that the class and venue type match as well as their sizes and the lecture/tut/lab, takes place in its respective faculty building. Figure 5 is the pseudocode of how the scoring function in the program works. It can be seen that no negative values are given and the lowest attainable score is, in fact, 0.

```
Calculate_Score():
    size_difference = Venue_Size - Class_Size

    if(size_difference == 0):
        score = 0
    else if(size_difference > 0):
        score += size_difference*0.2
    else:
        score += -size_difference*5

    if(Venue_Type != Class_Type):
        score *= 1.4
        satisfied_type = False
    else:
        satisfied_type = True

    if(Venue_Projector != Class_Projector):
        score *= 1.1

    score += distance**1.2
```

Fig. 5: Scoring Function

The score of an allocation is highly affected by the distance between the venue and the class' relative faculty building as well as the size difference between the allocated venue and class. This influences the score to favour allocations that efficiently accomdate students and is close to the class' respective faculty building.

F. Operators

There are four operators used in the scoring function. Each operator effects a different aspect of the network and involves swapping the class and venues of allocations in order to get a better score.

Variable	Description	Example
Course Code	Identification code used to distinguish between the different classes.	C2
Faculty	The faculty that the course belongs to.	S
Class Size	The number of students enrolled for the course.	200
Class Type	Classifies the classes into 3 class types: Tut, Lecture, Lab.	Lecture
Venue	Name of the Venue.	V1
Venue Size	The number of students that the venue can accommodate.	350
Venue Type	Classifies the venue into 3 venue types: Tut, Lecture, Lab.	Lecture
Distance	Matrix of distances from each class to each venue	243

TABLE I: Data Representation

1) Random Swaps between lower percent of scores: This offers a sense of randomness to the network which can allow the network to gain a better score. This swap is essential as the venues of the lower scored allocations can be fitting for the classes of the lower scored allocations.

2) Randomly Swap Highest and Lowest scored allocations: Swapping the venues and classes of the highest and lowest scored allocations allows for randomness in the high and low scored allocations. This can result in a better scoring network as a venue in the higher allocations can better accommodate a lower scored allocation resulting in a better scoring network.

3) Swap allocations where there is extra spaces with ones that are over booked: This swap allows venues with classes that possess extra space to be swapped with a venue that does not have enough space for the allocated class. This produces a network that better accommodates venues with classes. Here it can be seen that the venue and class type compromised as class-venue size accommodation is more valuable.

4) Swap allocations with venues that distant: This swap tries to allocate a venue that is closer to the class' faculty building. This is done by swapping the venue with another to allocate courses closer to their respective faculties. This allocation has an exponential weight allocated to it as the distance between classes and the respective faculty is highest priority.

G. Random Restarts

As mentioned before, the algorithm starts off with randomized allocations of venues and classes and then proceeds to perform operators on those allocations until the network score cannot be bettered. The algorithm produces a sub-optimum solution if not the optimum. To ensure that the best solution is found, the program utilizes random restarts which basically re-runs the program with a new random allocation list at the beginning. [Hu et al., 2009] These random restarts run a set amount of times. The best solution from all of these runs is the final solution returned by the program.

H. Limitations

The closest, most suitable venue may not always be allocated to a class as it may already be occupied. In the situation where all nearby venues are booked and the only venue for this class is far away (on another campus), the algorithm will select this venue as it is the only current and available solution.

I. Testing

1) Correctness: To test if the program produces a suitable allocation list, a small data-set of five elements will be input into the system. This will allow for easy checking if the output is correct, i.e. checking if the best venue is allocated to each class.

2) Analysis: The program is run on six data-sets, each increasing with size. The first data-set contains fifty venues and classes, the second contains a hundred and so on. Each data-set is run on the program with a thousand random restarts. The entire process is timed and also averaged over the one thousand random restarts. The results are recorded and analyzed thereafter.

V. Results and Future Improvements

In this section the results of the testing of the program as well as the future improvements of the system are discussed.

A. Program Correctness

The program was first run on a small data-set to examine how well the program performs on allocating suitable venues to classes. A data-set of five venues and classes were used to test this. Table II represents the data input into the program. Table III represents the best allocation produced from the program. It can be seen from Table III that the algorithm allocates the most suitable venue to each class.

It can also be seen (In Table III rows 2 and 5) that the algorithm favoured the closer venue rather than the one that would accommodate the five more students. This can be easily fixed by adjusting the weights of the distance and size attributes in the scoring function; however, for this research a closer venue is more favourable.

Venue	Venue Size	Course	Course Size	Distance	Score
V5	100	C2	245	87	1450
V4	80	C1	250	41	1395
V3	180	C4	80	93	493
V1	250	C3	180	90	464
V2	245	C5	100	36	220

TABLE II: Random allocations input to the system

Venue	Venue Size	Course	Course Size	Distance	Score
V4	80	C4	80	82	410
V2	245	C1	250	45	252
V3	180	C3	180	40	200
V5	100	C5	100	27	135
V1	250	C2	245	14	71

TABLE III: Recommended allocations output from the system

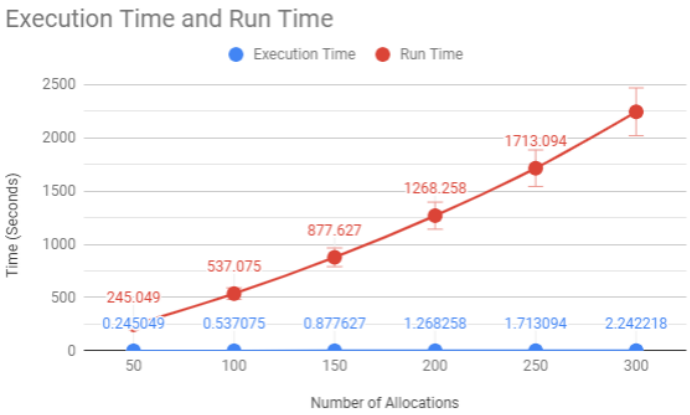


Fig. 6: The line graph illustrating the average time of a single execution and the total run time of each run.

B. Analysis

The program was run and tested as mentioned previously. The results are graphed in Figure 6. From Figure 6, it can be seen that there is an exponential trend. This was expected as there are more classes and venues with different attributes and sizes as well as distances to take into consideration. This means that there are many possible combinations of classes and venues. This solution of complexity $O(kn^2)$ is a significant improvement over a brute force approach to the problem which is $O(c!n!)$.

Using these results it can be estimated that the generation of all the best allocations in the university of Witwatersrand will take the best of a few hours. This is a reasonable amount of time since the output generated will be the most appropriate allocations in terms of distance and size. This result will allow other academic institutions to apply the same program in their own systems, allowing their institution to be working at an optimum rate.

C. Future Improvements

Currently, the system only takes in the distance of the venue to the class' respective faculty and how well the venue can accommodate the class as the main influences in the score calculation. This can be adjusted to take into consideration the class times, if there are night classes and day classes, as well as generate time tables that best suit students and staff members.

VI. Conclusion

The aim of this research project was to produce a more reasonable approach to the problem of venue allocation. This was accomplished using a BN to generate data and Bayesian Estimation to learn the parameters of the network. A custom scoring function was developed to score the allocation of a venue to a class in terms of distance to its respective faculty building and accommodation level of the venue for that class.

Thereafter, a *greedy* hill-climbing heuristic search was implemented to traverse the various alternate networks of allocations to find the network with the best allocation score. The developed system efficiently produced ideal lists of allocations, favouring closer venues over venues with better accommodation for its allocated class. This can be adjusted to favour better accommodation if needs be. The algorithm runs in a time complexity of $O(kn^2)$ as predicted. This time complexity is a major improvement over the previous complexity of $O(c!n!)$.

This system can be used by universities and other academia to allocate the best venues to classes in a short amount of time. In the future, the algorithm can be adapted to produce more intuitive time tables for universities and students alike. An alternate solution to the venue allocation problem is to use a graph-colouring algorithm to allocate the best venue to a class.

References

[Ajoodha and Rosman, 2018] Ajoodha, R. and Rosman, B. S. (2018). Learning the influence structure between partially observed stochastic processes using iot sensor data. AAAI.

[Christofides, 1976] Christofides, N. (1976). Worst-case analysis of a new heuristic for the travelling salesman problem. Technical report, Carnegie-Mellon Univ Pittsburgh Pa Management Sciences Research Group.

[Heckerman et al., 1995] Heckerman, D., Geiger, D., and Chickering, D. M. (1995). Learning bayesian networks: The combination of knowledge and statistical data. *Machine learning*, 20(3):197–243.

[Hu et al., 2009] Hu, X., Shonkwiler, R., and Spruill, M. C. (2009). Random restarts in global optimization.

[Khan and Iqbal, 2012] Khan, P. and Iqbal, M. (2012). Overcrowded classroom: A serious problem for teachers. *University of Science and Information Technology*, 49:10162–10165.

[Koller et al., 2009] Koller, D., Friedman, N., and Bach, F. (2009). *Probabilistic graphical models: principles and techniques*. MIT press.

[Munteanu and Cau, 2000] Munteanu, P. and Cau, D. (2000). Efficient score-based learning of equivalence classes of bayesian networks. In *European Conference on Principles of Data Mining and Knowledge Discovery*, pages 96–105. Springer.

[Pearl, 2014] Pearl, J. (2014). *Probabilistic reasoning in intelligent systems: networks of plausible inference*. Elsevier.

[Tsamardinos et al., 2006] Tsamardinos, I., Brown, L. E., and Aliferis, C. F. (2006). The max-min hill-climbing bayesian network structure learning algorithm. *Machine learning*, 65(1):31–78.