# Google Calender Reminder System

Edouard Eltherington 66389321 and Veronica Jack 12304986

12/12/2021

**GitHub Repository:** https://github.com/edouarde1/DBN-calendar-api

**Video Demo:** https://www.youtube.com/watch?v=BBLL8QbYvW4

## Introduction

The purpose of this project is to use a calendar API to build a personalized reminder system. We chose to use the Calendar API from Google to integrate our program with Google Calendar. The program grabs upcoming events and decides whether any reminder notifications are appropriate for each upcoming event depending on the user's profile (level of forgetfulness and if they are a night owl or early riser). The model also considers information about the event, such as the start time, priority level, and location. The system's decision-making process makes use of a Dynamic Bayes' Network model and utility function to decide how many, if any, reminders will be set in the user's Google Calendar App.

## Assumptions

1. The user uses Google Calendar
2. The user has their push notifications turned on for both the Google Calendar App and the device they are using Google Calendar on
3. When the user creates an event in their Google Calendar App, the user consistently:
   - Applies the color "Tomato" (Google's color code for red) for high priority events
   - Inputs a location (name or address) if they need to travel to the event
4. The number of reminders the user would like for an event are set as follows:
   - 1 reminder:
     - The reminder is set for 10 minutes before the event
   - 2 reminders:
     - The first reminder is set for 10 minutes before the event
     - The second reminder is set for 1 hour before the event.
   - 3 reminders:
     - The first reminder is set for 10 minutes before the event
     - The second reminder is set for 1 hour before the event
     - The third reminder is set for 1 day before the event
5. The user's sleeping habits do not change over time, but the user can recreate their profile with the program and indicate if their sleeping habits are different.
6. The user's forgetfulness changes slightly over time in that they become a little more forgetful over time.

## Model and Intuitions

Our motivation for a reminder system stemmed from the concept of needing preparation time before an event begins. This led to the following questions: Does the user need to travel to the event? Is the event of high priority? Is the user forgetful? Is the event starting at a time that is agreeable to the user's sleeping habits?

Putting these questions and motivation together, we created a conditional probability table (cpt) for the probability of needing preparation time given the event's location, event's priority level, the user's forgetfulness level, and the user's level of alertness.

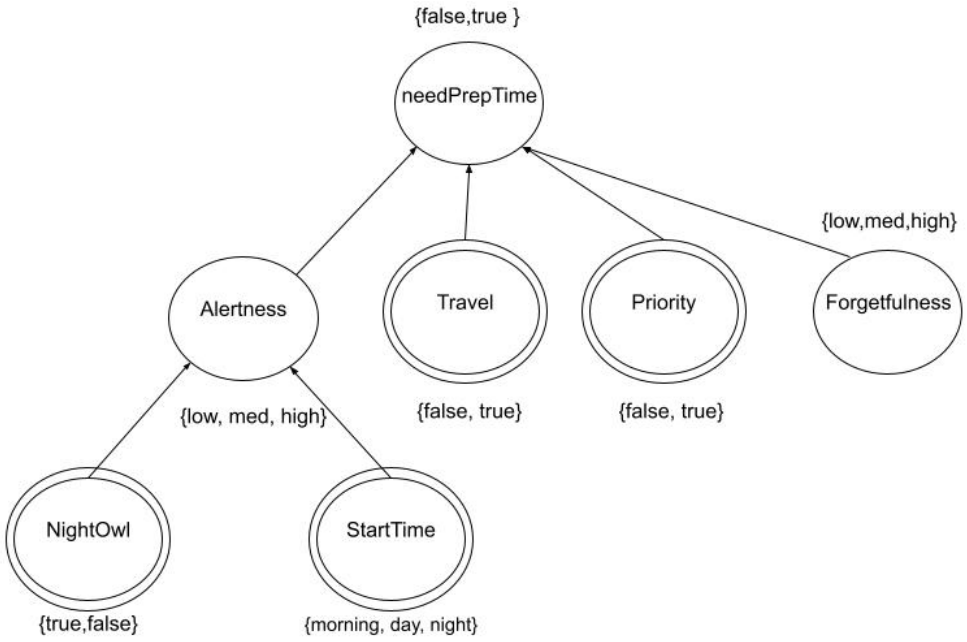



Figure 1: DBN Model

### Travel

If the user needs to travel to the event, which is indicated by having a name or location in the Google Calendar's location field for the event, then they probably need time to prepare and travel. If the user does not need to travel to the event, then the user probably does not need preparation time, or very little preparation time.

In an ideal scenario, if we had more time and resources at our disposal, we would change Travel from a binary variable to a more complex variable by using the location specified in the calendar, looking it up through Google Maps, and determining how much travel time would be necessary to get to the event with some extra time to ensure the user has preparation and travel time.

### Priority

If the event is a high-priority event, as specified by the user when they created the event in their Google Calendar and set the color to "Tomato", then the user probably needs a lot of time to prepare for the event. If the event is not a priority, then the user probably does not need preparation time.

Coming back to an ideal scenario, we would also add a layer to priority by looking at both the color set for the event and the name of the event. For example, if the event was colored to show that it is a high priority, but the name of the event included the words "exam" or "interview" then the preparation time would be increased compared to if the event name included "appointment" or "airport."

### Forgetfulness

The intuition behind this variable is that users who are more forgetful might need some extra time to prepare. Users who are not as forgetful will need less time to prepare for their events. As time passes, users will become more forgetful about their events.

One problem was trying to find a way to measure this variable. One idea that was discussed was figuring out user attendance at previous events. The calendar API would be able to detect if the user is making it to the events on time or at all. If the attendance to events is poor, then that is a good indication that the user is forgetful. On the other hand, if the user attendance is high, then the user was considered not forgetful. For reasons of scope, we did not include a way of measuring forgetfulness. Instead, during testing this value was fixed.

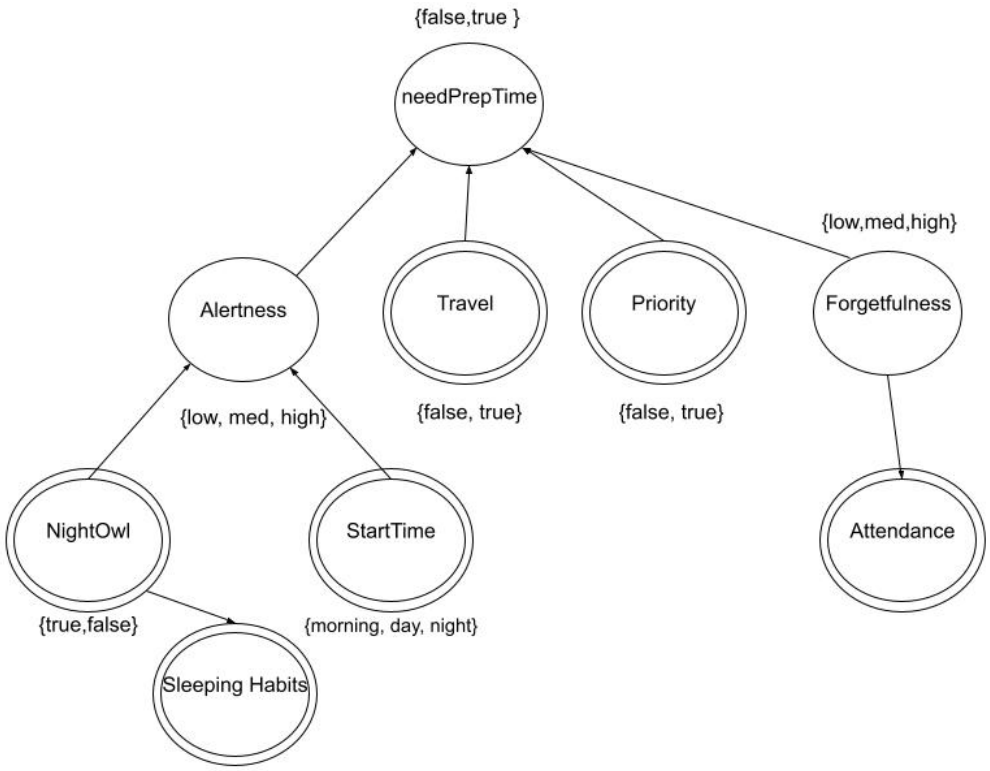
### Alertness

Our idea behind alertness is that it indicates a user's level of awareness of what is going on in the here and now. A user who is alert can respond quicker and remembers more, meaning they need less preparation time than someone who is not alert (they respond slower and have a harder time remembering in the moment).

This level of alertness is influenced by the event's start time and if the user is a night owl, meaning they tend to stay up late and wake up late). For example, if the event is in the morning and the user is a night owl, then the user probably needs some preparation time because their alertness would probably be low. If the event is during the day or night and the user is a night owl, then they probably do not need as much preparation time because they have a higher level of alertness. On the other hand, if the event is in the morning and the user is not a night owl but an early riser (meaning they tend to wake up early and go to bed early), then the user probably does not need as much preparation time because they are probably more alert.

## Ideal Graphical Model

As mentioned previously, we have many ideas that we would include if we had more resources and time. Unfortunately, we cannot implement our ideal graphical model, but we would like to share our vision of what this project could be.




# The Program

## Python

The python code will allow the user to access their google calendar, ask for user sleeping habits, and level of forgetfulness. Once the user enters these values, the python code will access the user's events information (Event, start time, priority, and travel information). The event information, along with sleeping habits and user level of forgetfulness, will be sent to the DBN in matlab. The model will spit a value that indicates the best number of reminders to set for a given event. The python code will then adjust the number of reminders for the event.

## Matlab

The Matlab portion of the program creates a Dynamic Bayes' Net Model based on our intuitions, and then uses an expected utility function to simulate the scenario of the event over multiple time steps to determine which action tends to be the best one.

# Simualtion Examples

The following plots give a visual representation of how the model behaves under test conditions. The three plots show the outputs of the expected utility change with the observed nodes (travel, startime, priority, and sleeping habits), over a series of timesteps. In each case there will be a different number of reminders added to the event following our intuitions.

## Case 1: Three Reminders

Phineas is a user of our calendar api and has already created a profile in the app that indicates that he is a night owl. He creates an event called "Airport Check-in", flags it as a priority by changing its color to "Tomato", sets the start time to 8:00 am (when he needs to arrive at the airport), and inputs "Airport" as the location to show our reminder system that he needs time to travel. Following our intuitions, Phineas would need many reminders.

After the application creates the DBN model, it runs the simulation and sets the following variables:

- isNightOwl = true (2)
- startTime= morning (1)
- travel = true (2)
- priority = true (2)

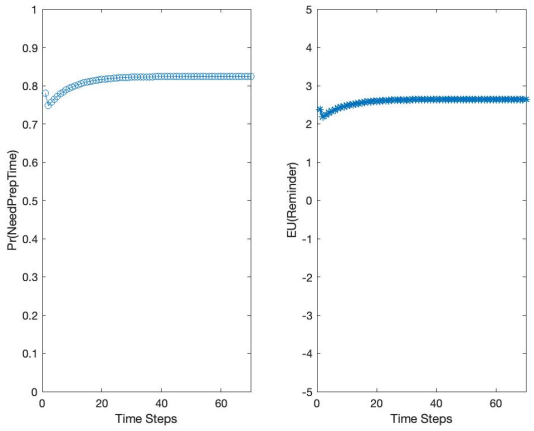



Figure 2: DBN Plot for Three Reminders

In this plot, the probability that Phineas needs preparation time at t=0 in the simulation is ~82%. The plot shows that Pr(needPrepTime) increases over a series of timesteps. This increase comes from our transition function for "Forgetfulness". As the time moves on the user becomes slightly more forgetful, so the user will need more time to prepare.

The end result for the expected utility of setting a reminder is around 2.8, which indicates that the best action is to give three reminders because 2.8 exceeds our intuition threshold. The first reminder will be set the previous day, the second will be set 1 hour before the event, and the third reminder will be set 10 minutes before the event.

The result is reasonable from the intuitions of the model. This event is a priority, Phineas will have low alertness since the event is in the morning and he is a night owl, and he will need time to travel. All three reminders will ensure that he arrives at his event on time.

## Case 2: No Reminders

Phineas is now on vacation and is spending long relaxing days in Paris. He is planning to walk around the Louvre at 6:00 pm, so he enters it into his Google Calendar. Considering that this is a leisure activity, Phineas does not think this event is a priority, so he does not change the color of the event to "Tomato." The event is in the evening, and no travel time is needed, so Phineas leaves the location blank in his Google Calendar. Based on our intuitions, Phineas will probably not need any reminders!

- isNightOwl = true (2)
- startTime = night (3)
- travel = false (1)
- priority = false (1)

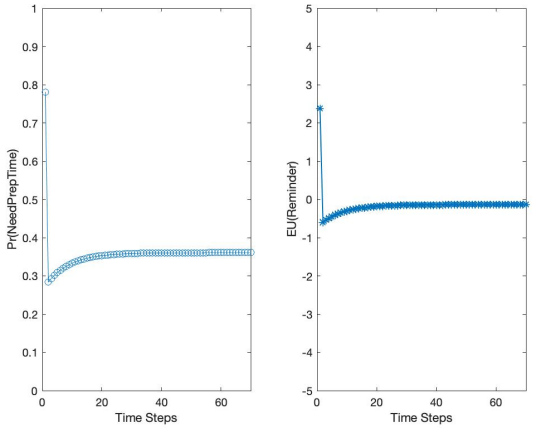



Figure 3: DBN Plot for No Reminders

In the simulation, the probability of Phineas needing preparation time is initially set to ~29%. As the time steps increase, the Pr(needPrepTime) increases. This increase of probability comes from the transition function for forgetfulness. The expected utility for setting a reminder is around -0.2, which results in the best action to be no reminders.

The scenario is ideal for Phineas because he will be very alert before the event, since it's in the evening and he usually likes to stay up at night. In addition, for an event that is not a priority, it would be frustrating for him to be receiving constant reminders.

## Case 3: One Reminder

When Phineas returns from his trip, he plans to meet with Ferb for lunch to tell him all about his trip! He might be a bit tired since he does stay up late, and the event is in the daytime. Although, this event is not a priority for Phineas and he won't need to travel since the restaurant is right below his apartment. It would be nice to get at least one reminder for this event knowing that his alertness might not be on the low side.

After Phineas adds his lunch event to his Google Calendar, he runs our reminder system app, and it sets up the simulations to determine the best number of reminders. For this simulation, the following variables are set:

- isNightOwl = true (2)

- startTime = day (2)
- travel = false (1)
- priority = false (1)

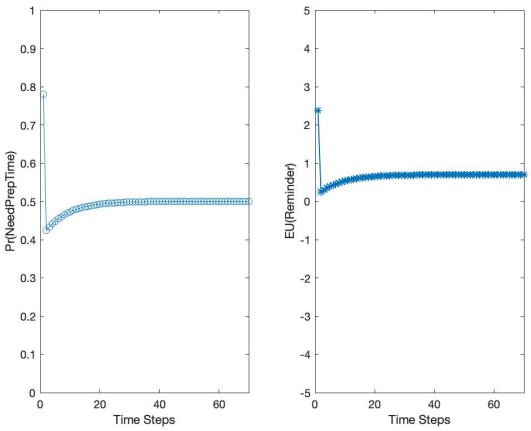



Figure 4: DBN Plot for One Reminder

In this plot Pr(NeedPrepTime) is initially at ~41% which is moderate. Similar to the above plots the Pr(NeedPrepTime) will increase as the level of forgetfulness increases over each time step. The EU reaches to about 0.8 which results in there being at least one reminder for this event.

The scenario is also nice for Phineas since he will have a reminder 10 minutes before Ferb arrives at the restaurant, which will get him ready and out the door on time!

## How to Test

If you have downloaded our program and completed the prerequisites defined in the README, you can run simulations in two different ways.

The first option is to run the program as intended. When the program runs for the first time, it asks for the user's information and stores their profile as a text file. If the user_profile.txt file is deleted, the program will run as if it was the first time and ask for user information.

The second option for running simulations is to run the Matlab code by itself. After opening a Matlab session, the user can run the following code:

- init.m
- dbn = mk_needPrepTime
- ex = 1 % 1: random, 2: fixed observable
- isNightOwl = 1 % 1:false, 2:true
- startTime = 1 % 1:morning, 2:day, 3:night
- travel = 1 % 1:false, 2:true
- priority = 1 % 1:false, 2:true
- sim_decision(dbn, ex, isNightOwl, startTime, travel, priority)

## Future Development

This DBN captures a variety of situations that would cause an event to have a certain number of reminders based upon sleeping habits of the user, travel time, startime, and priority of an event. Given these inputs, the model predicts a number of reminders that would be most suited for each event. Having a basic model provides a simple way of changing event reminders over time steps. However, areas could use further development.

More temporal relationships would more accurately reflect probabilities of event preparation times. In this model, forgetfulness is the only transition function. As seen with the graphs, this causes the initial probability of needPrepTime to remain steady over each time step. Once the model predicts the number of reminders in the first few time steps, the best action does not change. Adding in more variables with a temporal relationship could improve the way it sets reminders over many timesteps.

In the process of making the model one variable that was discussed was time until an event starts. The variable in real life has a strong effect on how many reminders a user would need for an event. Imagine creating an event a couple months down the line, this would require more reminding! Although, keeping track of when a user creates an event and the inutions behind this variable were out of the scope for this project.

As mentioned for our ideal graphical model, we would like to convert the binary travel variable into a more complex variable that would use Google Maps to estimate required travel time. In addition, we would prefer to havee more observable variables. These variables would influence Forgetfulness and NightOwl depeding on the user's attendance (if they attend every event and if they are late to events). NightOwl could be influenced by how many events tend to be at different times of day.

Lastly, this model relies on users being consistent with their reminders. Users might not always mark their events as a priority or might forget to include the event location. One idea would be to include some text analysis on the event name to automatically class the priority level and intuitively determine if travel time is required if a previous event of the same name had a location set. This would reduce the reliance on the user and create better predictions for event reminders.

## Incomplete Features

Two features were not completed for this project, as they were deemed unnecessary for the main goal of this project.

The first incomplete feature is the third setting for the simulation environment which allows the user to test the DBN by fixing hidden variables. This was not implemented because the results from the first two simulations settings indicated that the program could successfully complete the project goal.

The second incomplete feature is using the input from the user for their level of forgetfulness. Instead of using this input as evidence in the DBN, we changed forgetfulness to be a hidden variable that gets slightly worse over time. As mentioned in our future development section, forgetfulness would ideally be influenced by other observable variables based on user attendance.

## Conclusion

Using the Bayes' Net Toolbox and Google Calendar API, we developed a personalized Google Calendar reminder system. This system pulls upcoming events and determines the best number of reminders to set in the user's Google Calendar using 7 variables (3 hidden) The event variables involved in this process include the event start time, priority level, and if the user must travel to the event. The user variables are their level of forgetfulness, if the user needs prepration time, and the user's level of alertness based on their sleeping

habits. The system's decision-making process makes use of a utility function to decide to either set 0, 1, 2, or 3 reminders at set intervals before the event begins.

For the DBN model, we handcrafted our CPTs based on our intuitions, and then we created two simulation environments to test the model over time. We use a built-in inference algorithm to update our belief distribution and plot the probability of needing preparation time and the expected utility of setting a reminder over time. The best action is then retrieved from these simulations and is incorporated into the calendar API.

This program provides a solid base for future developments that could use more information about the user from their calendar (and potentially their location, depending on user preferences) to implement more observable variables to better estimate if the user needs preparation time before an event.

## Acknowledgements

We want to thank Dr. Bowen Hui for helping us convert our intuitions into an accurate graphical model. We also thank Logan Parker, and Rylan Cox for their assistance in debugging our Matlab code that implemented the Bayes' Net Toolbox.