# Adaptive Nested Algorithms for Balanced Scheduling

Stetson Bost

## Introduction

Scheduling is very important for staying organized and managing stress for both individuals and organizations of all kinds. For this project, we developed algorithms that create effective, adaptable, and balanced schedules.

## Algorithm for Personal Scheduling

Our nested algorithm for creating personal schedules aims to mimic human behavior using machine learning to schedule tasks with deadlines while maintaining a balanced schedule. We define a *schedule* to be a collection of relatively short *time slots*, with contiguous time slots forming *blocks*. We define a *section* to be a group of blocks. Our algorithm has three nested steps.

1. Assign tasks to sections, prioritizing tasks according to their deadlines so that all tasks are completed on time, if possible.
2. Assign tasks to blocks within sections, prioritizing a balance of intensity levels among blocks.
3. Assign tasks to time slots within blocks in a way that can adapt to personal needs and preferences.

Each step takes into account various aspects of the tasks, including deadlines, estimated duration, and intensity. Additionally, each block has a minimum permissible amount of time reserved for non-work activities. This can be used for structured activities, such as doing laundry or preparing a meal, or it may be unstructured personal time, depending on user preference. By maintaining similar intensity levels among blocks, the algorithm favors a balanced schedule.

| Task | Deadline | Duration | Intensity |
|---|---|---|---|
| Task 1 | Monday | 6 hours | 0.3 |
| Task 2 | Friday | 3 hours | 0.6 |
| Task 3 | Thursday | 4 hours | 0.5 |
| … | … | … | … |

**(a)** Start with a list of tasks that need to be completed. Each task has a deadline, estimated duration, and intensity level, where a higher intensity level corresponds to a more difficult or stressful task.

Section: Tuesday 8 AM - Midnight

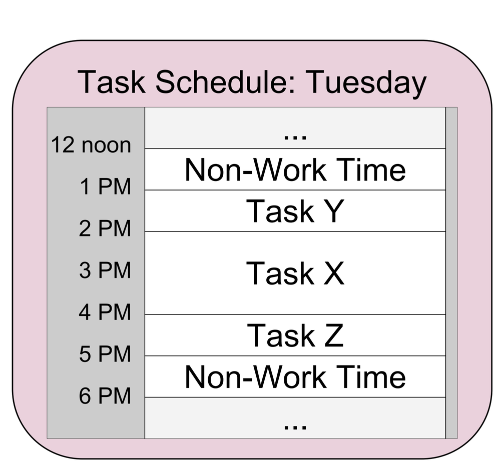
| | | | |
|---|---|---|---|
| Block 1 | 8 AM - 12 noon | Non-Work Time | 4 hours |
| Block 2 | 12 noon - 6 PM | Task V | 2 hours |
| Block 3 | 6 PM - 11 PM | Task W | 1 hour |
| | | Task X | 2 hours |
| | | Task Y | 5 hours |
| | | Task Z | 2 hours |

**(b)** Step 1: Create a section made up of time blocks, and assign tasks to the section, prioritizing tasks with the soonest deadlines. Some amount of non-work time is associated with each block, and we treat non-work time as a task.

Block 2: Tuesday 12 noon - 6 PM

| | |
|---|---|
| Non-Work Time | 2 hours |
| Task X | 2 hours |
| Task Y | 1 hour |
| Task Z | 1 hour |

**(c)** Step 2: Assign tasks to each block in the section, prioritizing a balance of intensity levels from block to block.




**(d)** Step 3: Schedule tasks within the block. This step can be done in any way the user prefers.

**Figure 1:** Example of the personal scheduling algorithm, focusing on a section containing three blocks on Tuesday.

## For Further Information

For more information about this project, please contact `sbost@hmc.edu` or `gu@hmc.edu`.

## Algorithm for Event Scheduling

We developed two algorithms for scheduling events within groups. Members of a group often have to arrange events where a number of people must attend. An event may be either recurring ("regular") or non-recurring ("special"), may have restrictions on available times, and may have a list of desired or required attendees.

### Probabilistic Approach

We created an algorithm for event scheduling that uses historic data and a probabilistic version of the $k$-nearest neighbors algorithm to determine the likelihoods that all potential attendees will attend a specific event scheduled at a given time (Holmes and Adams, 2002). By summing these likelihoods across all attendees, we are able to get the expected value for the number of people at the event.

### Nested Approach

We also developed a nested algorithm that makes use of the nested personal scheduling algorithm detailed to the left with some slight modifications. The basic structure of time slots, blocks, and sections is the same, but it may be less likely to have adjacent blocks or contiguous sections, depending on the group's scheduling practices. We treat events as tasks with possible time windows and use those to create sections. In place of task intensity, we use a quantity that describes the priority of the event. We also introduce some conditions for balance, such as restrictions on the frequency of regular and special events.

## Discussion and Future Work

We are in the process of running the algorithms with a large data set.

In the future, these algorithms could be used with individuals and groups that are willing to try schedules that are produced. Additionally, for an organization that uses a common calendar application, there may be potential to integrate the event scheduling algorithms with the calendar to automatically schedule events.

## References

Holmes, C. C. and Adams, N. M. (2002) A probabilistic nearest neighbour method for statistical pattern recognition. *J. R. Statist. Soc.* **64**, Part 2, pp. 295–306. `http://hedibert.org/wp-content/uploads/2016/02/holmes-adams-2002.pdf`.

## Acknowledgments

I would like to thank my advisor, Professor Weiqing Gu, for her guidance throughout this research. I am also very grateful for the kindness and generosity of the Rose Hills Foundation through their Science and Engineering Summer Undergraduate Research Fellowship, which has made this project possible.

**Advisor:** Weiqing Gu