Event Scheduling with Soft Constraints and On-Demand Re-Optimization

Patrick Hosein and Steffan Boodhoo
Department of Computer Science
The University of the West Indies, St. Augustine, Trinidad
e-mail:patrick.hosein@sta.uwi.edu,boodhoo100@gmail.com

***Abstract*—With the increased use of today's information technologies, more people are using electronic calendaring systems, such as Outlook, to schedule everyday events. This includes work related items such as meetings as well as personal items such as birthdays and their kids' events. There have been no significant changes to this process. People enter their personal event items on their calendars. If someone needs to schedule a meeting, they will view available time slots of all potential attendees and choose an appropriate date and time. There are two weaknesses in this process that we address. Firstly, the value of an event to a person is not specified and hence when scheduling a meeting, one simply assumes that any slot that is marked busy is unavailable for the meeting. However, a busy slot could be for something that is not as important as the meeting to be scheduled. The second weakness is that once a meeting is scheduled, the chosen time slots are not changed unless the organizer decides to do so for some specific reason. However, the optimal time slot for the meeting may change over time. In this paper, we propose a framework that addresses these issues and illustrate its value through simulations.**

Keywords-Decision Support System; Electronic Calendar; Optimization; Event Scheduler

I. Introduction

With the advent of computers and the Internet, automated event and meeting scheduling has taken on a greater role. However, certain aspects of traditional scheduling have not changed. In particular, time slots for a user are typically designated as busy or non-busy, and an event can be scheduled in a free slot but not in a busy one. However, the value of a slot to a user can vary widely since it may be used for an important doctor's appointment or for something less important such as a gym class. We believe that more appropriate schedules can be obtained if this finer granularity of information is made available. The specific reason why a slot is of high importance to a user need not be disclosed since such information may be confidential. Another issue with traditional approaches is that, once an event is scheduled, it remains in those time slots and future (possibly more important meetings) must be scheduled around it. This may limit the number of meetings that can be scheduled over a given time period. Our prime contribution is the introduction of a framework for scheduling events that (a) includes soft constraints (i.e., allocation of a value of a time slot by a user rather than a yes/no availability constraint) and (b) the ability to re-optimize schedules if significant benefit can be gained by the affected users. We next describe the related work in this area followed by a mathematical formulation of the problem and then numerical results.

A. Related Work and Contributions

The models used in the literature in this area mainly follow the traditional approach of hard constraints and hence there are no similar models with which to compare. There are some publications that discuss re-optimization but using different approaches. Most related work were found in patent filings.

The paper by [1] considered the problem of scheduling staff, and so in their case a specific problem was addressed. Furthermore, they used hard availability constraints and the traditional sequential scheduling. The paper by [2] looks at personal time optimization, which is similar to what we are doing, but they only address the schedule of a single user and used Artificial Intelligence techniques. Of course there are the mainstream approaches such as Outlook from Microsoft [3] and Google Calendar from Google [4], but as previously mentioned these use standard methods for user availability.

In the patent [5], the inventors consider the case of scheduling meetings, but optimize based on non-standard criterion. The objective could comprise of multiple criterion, such as; travel time, minimizing a wait time, maximizing a contiguous meeting time, choosing a preferred meeting location, minimizing a variance from a preferred meeting time, maximizing the attendees who are desirable, or scheduling the calendar event as close as possible to a target date. The patent [6] can be described as an Internet based version of Outlook. The patent [7] is again traditional, but allows for the relaxation of constraints in the case that an event cannot be scheduled. Patent [8] allows participants to code actions to be taken in case a meeting time cannot be obtained. An action, for example, could mean leaving early and so allowing for a shorter meeting. Patents [9] and [10] are both quite similar to Outlook but allows for a best fit allocation, in case an event cannot be scheduled. Finally, the patent [11] allows one to insert flexible events. These are events that can be moved around if it conflicts with a new event that has no feasible solution. The degree of conflict is taken into account. The patent [12] is similar to our approach in that they consider soft constraints. However, their solution is client based (i.e., each user solves their scheduling independently), while ours is server based and the optimization is performed over all participants simultaneously, and hence a global solution is obtained.

Our main contributions are the inclusion of soft constraints for availability and a framework for re-optimization of schedules. New schedules are only used if they provide sufficient gain over the present solution. We illustrate the gains through simulations.

II. Proposed Framework

We consider a sequence of time slots, each of duration τ. The total number of slots considered will be denoted by T. For example, suppose that $\tau = 0.25$hrs and we want to consider a

978-1-5090-3469-7/16/$31.00 ©2016 IEEE

week of time slots, then $T = 7 \times 24 \times 4 = 672$. Time slots will be indexed by t. We assume N people for whom optimal time schedules are to be computed. An event (such as a meeting) extends for an integer number of successive time slots. We define two types of events, personal events are in the complete control of the person while public events involve two or more people who may or may not decide to participate in the event. For example, a personal event may be a doctor's appointment while a public event may be a department meeting involving all members of a department. The set of participants of event k will be denoted by $\mathcal{E}_k$ and the range of time slots within which the event can be scheduled will be denoted by $\mathcal{R}_k$. We assume that, for each time slot, a person can participate in at most one event (public or personal).

A. Soft and Hard Constraints

In traditional calendars, a person is either busy or free in each time slot. If an event consisting of two time slots is to be scheduled, then one must find two successive slots for which all event participants are free. This can sometimes be difficult if many people are participating in the event, or if several of the participants are very busy. We consider such constraints to be hard constraints since if one indicates that they are busy in a slot we assume that slot cannot be used.

We introduce the notion of soft constraints for events. Here one indicates the relative value (on a scale from 0 to 10) of the event to the person. For example, a doctor's appointment has a high value since missing it can lead to paying a fee, having to reschedule, and delays in the required remedy. On the other hand, if the event is going to the Gym then the value is significantly less since it may be easy to go at a different hour. If hard constraints are used then both of these events will be treated the same when scheduling. However, we believe that they should not which is why we instead allow soft constraints.

For user n, if a time slot contains an event, then a value v_n is assigned to the time slot and this corresponds to the value of the corresponding event. If no event is scheduled for a slot one may still assign a value which would correspond to how important it is to the person to just have some free time. If the event is extremely important (e.g., a surgery) then a value close to 10 is assigned. In general some value between 0 and 10 is assigned to each event and in turn assigned to each slot of that event. Values for personal events are assigned by the user. The value of a meeting event is decided by the meeting organizer but we will discuss variations of this approach later. An event can only replace another event if its value is greater.

Given a set of users and a set of personal events we address the problem of optimally scheduling a given set of public events. Let us first discuss this problem in the manner in which it is presently addressed whereby events are scheduled in the order in which they are created. So, assume we have previously scheduled events and decide to schedule a new one. For a given set of consecutive time slots, the objective is to find the necessary number of consecutive time slots for the event such that the total value gained by adding this event is maximized. Not all users may attend the event since a user will only attend if the value of the event exceeds the value of whichever event is presently in that time slot.

The value chosen for an event is very important since users will value personal events relative to typical meeting events. For example, if a department meeting is valued at 0.8 then a member of the department should not value a gym class higher than this. On the other hand they may value a doctor's appointment with a value above 0.8. Later we will discuss how meeting values should be chosen. The same approach can be used for personal events. For a new event one can then simply evaluate the sum value for each potential slot allocations and then choose the one with the largest sum value. When events are canceled then the corresponding slot(s) become free and assigned the default value.

B. Value Maximization

As we add events, the overall objective is to increase the total value of all users over all slots. We saw above that the total value is non-decreasing as a function of number of events since each event results in a larger value for a participant (if the event is accepted) or no change. However, this sequential allocation may not be globally optimal. In other words if we simultaneously allocate all future events we may in fact get a better solution. Let us illustrate this with a simple illustrative example.

Consider two users and two time slots. The initial values (e.g., based on personal events) for these users are [0, 0.1] and [0.1, 0.9]. Consider two events each requiring one slot and with a value 0.2 for the first event and 0.8 for the second. If we were to schedule event one first then it would be placed in slot 1 since it provides the largest value gain. The user values now become [0.2, 0.1] and [0.2, 0.9]. Now if we schedule the second event then it would also be placed in slot 1 and the user values now become [0.8, 0.1] and [0.8, 0.9] for a total value of 2.6. Now suppose we instead scheduled the second event first followed by the first event. The second event would be placed in slot 1 to give [0.8, 0.1] and [0.8, 0.9]. If we now schedule the first event it will be placed in slot 2 to give [0.8, 0.2], [0.8, 0.9] with a total value of 2.7. Therefore if the second event had come in first the resultant schedule would have been better. In general one can obtain better schedules by scheduling all events together rather than by doing one at a time. Although better schedules can be obtained by rescheduling multiple events scheduled in the past, making such changes can be annoying. Therefore we will look at other, more user friendly ways, to improve performance.

When we formulate the mathematical problem we will see that the determination of the optimal solution can be very compute intensive especially for large problems. A complete re-optimization may also not be desirable since it may cause changes in many meetings and all attendees will have to be rescheduled. However such re-optimization can result in significant improvements especially when previously scheduled events are canceled by the organizer. In the next section we provide a mathematical model that we use for performance evaluation. This is followed by simulation results where we illustrate the benefits of our two contributions, soft constraints and on-demand re-optimization.

III. Mathematical Model

In this section we present the mathematical formulation of the global optimization problem. We define the following:

N = the number of participants, indexed by n,
T = the total number of time slots considered, indexed by t,
K = the total number of public events, indexed by k,
$\mathcal{E}_k$ = the set of participants to be scheduled in event k,
v_{nt} = value of a personal event scheduled by user n in slot t,
V_k = value of event k in any of its scheduled time slots,

$\mathcal{R}_k$ = range of possible starting slots for event k,
d_k = the duration of event k in slots,
s_k = first slot assigned to event k (the decision variable).
$\vec{s}$ = $\{s_1, s_2, \ldots, s_K\}$

v_{nt} are values for personal events and these can be changed by a participant at will. If a participant decides to attend event k in that time slot then this value is replaced with the value of the event V_k. For each time slot a participant chooses the event (personal or not) that has the maximum value. Of course this must be an event to which the participant was invited. The objective is therefore to schedule events so as to maximize the total value of all users over all time slots. The decision variable is s_k which is the slot chosen as the first slot of event k. Also note that we allow the overlap of events.

One drawback of this approach is that a participant may end up partially attending one or more events, or a personal event may be chosen within the range of a scheduled public event. These issues occur in real life. In the case of overlapping public events the user can use other personal factors to determine which event is more desirable. He/She would then indicate non-attendance of the other event to the organizer. In the case in which a personal event has been chosen within a public event, the user must again decide which to choose. If the personal event is chosen then the organizer of the public event is notified otherwise if the public event is chosen the value of the personal event is reduced to a value below that of the public event. The optimization problem can therefore be stated as follows:

$$\max_{\vec{s}} F \equiv \sum_{n=1}^{N} \sum_{t=1}^{T} \max \left\{ v_{nt}, \max_{\{k | n \in \mathcal{E}_k,\ s_k \le t \le s_k + d_k\}} V_k \right\} \quad (1)$$

$$\text{s.t.} \qquad s_k \in \mathcal{R}_k \qquad \forall k$$

This is explained as follows. The function $F(\vec{s})$ is the total value of all users over all time slots. If no events are scheduled for a particular time slot for a user then we use the value of any personal event in that slot for that user. However if one or more events have been scheduled for that user in that slot then we must determine the new value if the user decides to attend one of these events. If the user decides not to attend any of these events (because of their lower value than any personal event in the slot) then the slot value remains the value of the personal event. Otherwise we consider all events for which the participant is scheduled. Within this set we find the event with the largest value over all events that have been scheduled in this time slot. We then assign the value of this event to the time slot for the user. Note that the start time of an event is restricted so we include this as a constraint.

IV. Solution Methods

In this section we look at some approaches to solving this optimization problem. We then present numerical results to illustrate which of these approaches may be suitable for implementation in a real system.

A. Global Optimal Solution (OS)

Each time a new event is received we can re-optimize all schedules (i.e. the global optimization problem). However this has two major drawbacks. In order to do this we have to look at all possible combinations of event schedules to obtain the best solution. This requires $\prod_{k=1}^{K} |\mathcal{R}_k|$ schedule evaluations which can be quite time consuming. Another issue with this approach is that each time a new event is received we may potentially have to ask a significant number of participants to change events that were previously scheduled and this will be frustrating. Due to computational costs it was impractical to simulate realistic cases with this method, therefore we do not recommend this approach.

B. Traditional Sequential Solution (TS)

For all invitees, each day and potential timeslot is evaluated where a potential timeslot is one where the index of the timeslot in addition to the length of the event does not exceed the day. Time-slot evaluation is the process by which each timeslot from the potential timeslot to potential timeslot plus event duration is compared against the events weight. If the weight is greater than the time-slots then we add the difference. The sum of differences for an event across all invitees for a potential timeslot represents the value added by choosing this timeslot for the current event. After going through all potential timeslots we place the event where there exists the largest sum of differences, that is, where users had the least important timeslots. When a timeslot is chosen invitees and their timeslots are updated accordingly. Unfortunately, as we have seen previously, the schedule obtained by this sequential approach may be quite sub-optimal.

C. Traditional Solution with Hard Constraints (HS)

For this approach two copies of the users schedules were created, the first was converted to binary and used as logic, the second was used to keep track of actual values. Since hard constraints are either available or busy, the first copy was converted to binary values where any user event valued from 0 to 2 is mapped to 0 (i.e. available) while the rest are mapped to 1 (busy). The process was then similar to TS in that HS iterates through all potential timeslots but only counts the number of 0s. The timeslot with the most 0s among invitees during the current events life was chosen, that is, the timeslot when users are most free during the event. When a timeslot is chosen, for each invited person and each timeslot in the event we set the 0 to 1 in the first copy and in the second copy we update the user event value. So the binary list is used for scheduling but the event value list is used to evaluate the performance of the resulting schedule.

D. Coordinated Ascent Solution (CS)

In this case we first find the traditional solution (using TS) but then, we do the following. We pick one event at a time and, keeping all other schedules fixed, we re-optimize for that event. The sequence of events used for this case is the same as the sequence in which they were generated. If we consider an event as one degree of freedom (a coordinate) then this consists of optimizing along one coordinate direction at a time hence the name. If each event is re-optimized then the number of schedule evaluations will be $\sum_{k=1}^{K} |\mathcal{R}_k|$. This complexity lies between that of the optimal approach and the sequential approach but provides an objective value close to optimal.

V. Numerical Results

In this section we provide numerical results to better understand the practicality of using this type of framework. Note, however, that we do not include input from users when there is an overlap in their events (public or private).

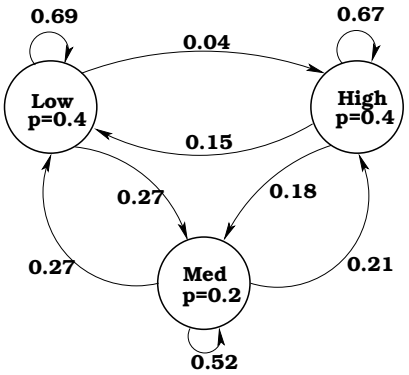

Figure 1. Markov Model for Personal Event Value Determination

A. Baseline Problem

We used the following approach to generate data. We requested scheduling data from a large group of people. They were asked to provide personal event values from 0-10 for 12 timeslots corresponding to one hour slots from 7am to 7pm. The values were grouped into three states 0-2 (low), 3-7 (medium) and 8-10 (high). After collection of questionnaires and removal of outliers the responses were tallied. Using this data the Markov model presented in 1 was developed and used to generate data for our experimental results.

For each public event, the number of participants, and the length (in timeslots) were each randomly generated within a set range, and their values were randomly generated between 5 and 10 as public events tend to be more important than personal events. Participants are randomly selected from the user base according to the previously generated number of participants. We start with a baseline problem and we then vary individual parameters (number of events and number of timeslots) to investigate how performance varies with these parameters. The parameters for the baseline are as follows:

N = 100,
T = 18 hours x 5 days = 90,
K = 50,
$\mathcal{E}_k$ = randomly pick $p \in [0, 1]$, choose user with probability p
V_k = randomly pick from 5 to 10
v_{nt} = generated using the Markov Model
d_k = chosen randomly from 1 to 9 (maximum half day)
$\mathcal{R}_k$ = from slot 1 to slot 18-d_k each day of the week

B. Performance Results

We first show the performance benefits of the proposed approach by solving the baseline problem but with 500 events. We generated 100 sample problems and for each we computed the total user value over all slots. We then averaged this over all 100 runs. In addition to collecting value information we also computed the fraction of users who attend each event and average this over all events in each run. This provides an idea of how well the scheduled events are attended which is also of importance. In order to determine the degree of fairness of the allocations we compute the Coefficient of Variation of user values in each run and average this over all runs. The coefficient of variation is the ratio of the Standard Deviation and the Mean. Values close to 0 imply that all users achieve similar average values. We average collected statistics over all runs and normalize value and attendance with the traditional approach. This information is presented in Table I.

Table I. PERFORMANCE RESULTS FOR THE BASELINE PROBLEM

Algorithm	Normalized Value	Normalized Attendance	Average CoE
HS	1.0	1.0	0.12
TS	1.4	3.8	0.08
CS	1.8	5.2	0.03

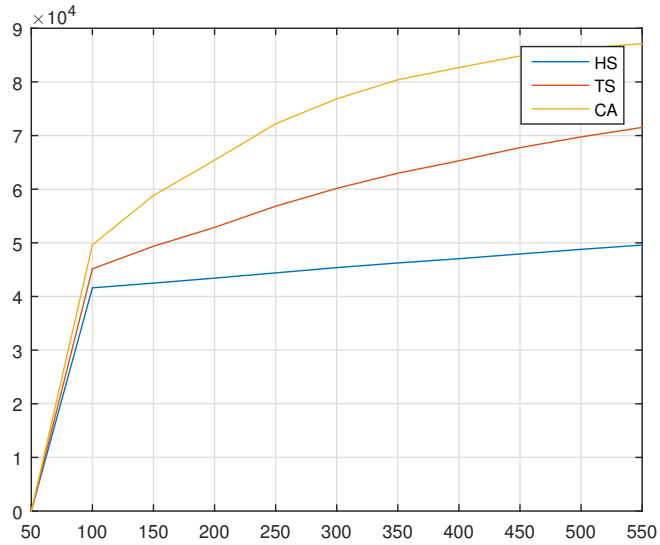

Figure 2. Change in Value as K varies from 50 to 500

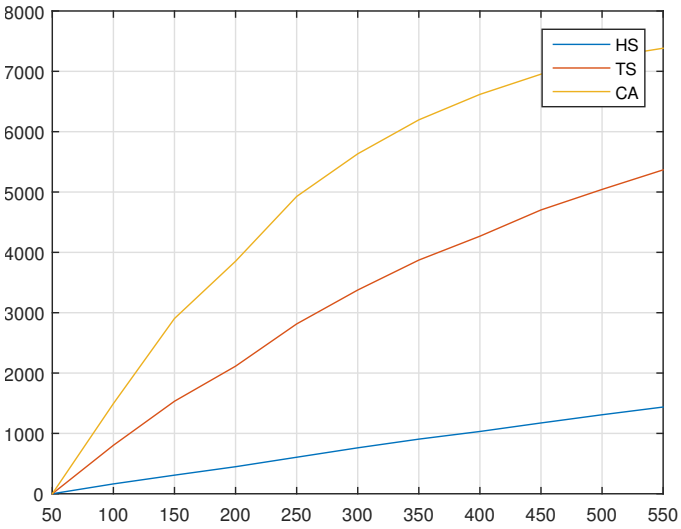

Figure 3. Change in Attendance as K varies from 50 to 500

We observed that the introduction of soft constraints with no re-optimization (TS) results in significant gains. Furthermore, the addition of re-optimization to TS (CS) results in even more gains so both proposed additions (soft constraints and re-optimization) are clearly beneficial. Next we investigate the sensitivity of these performance results to the various parameters. We vary various parameters and plot the computed metrics versus the value of the parameter. We vary the number of timeslots (by varying the number of days) from 10 to 100

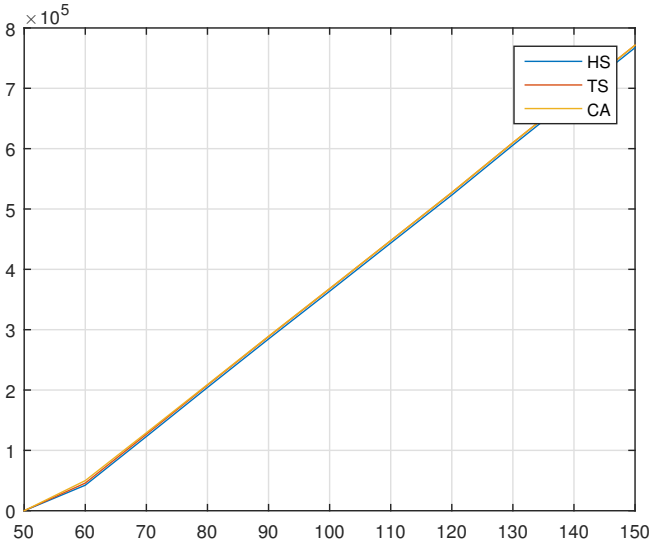

Figure 4. Change in Value as T varies from 10 to 100

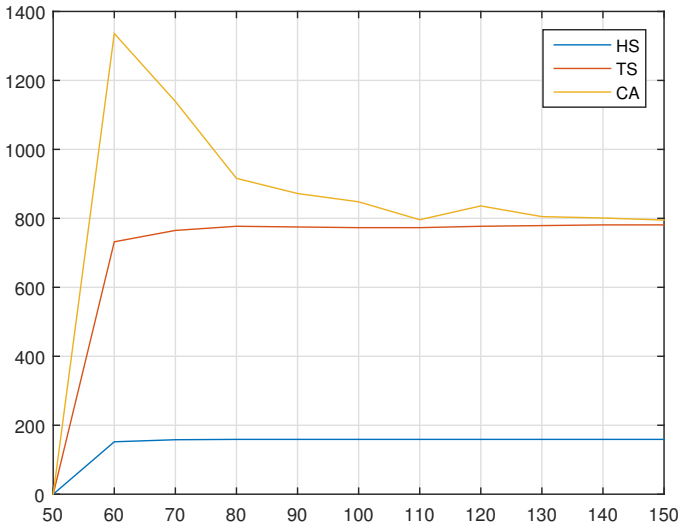

Figure 5. Change in Attendance as T varies from 10 to 100

We observed the performance advantage of the proposed approach increases with events since it is better able to allocate slots especially as the system gets busy. We see little dependence on duration of the scheduling space but this is due to the fact that the number of events was small and we know under such conditions all approaches can find suitable schedules. What is interesting is the significant gains in attendance. This is due to the re-optimization process and hence we see the need to globally optimize events.

VI. Summary and Future Work

In this paper we presented a framework for including values to events in an electronic calendar and we demonstrated the potential value this provides. We also evaluated the gains possible by performing re-optimization. Here we see even bigger performance gains. We plan to deploy a prototype system and obtain feedback from users to see how such a framework performs in real life.

References

[1] A. T. Ernst, H. Jiang, M. Krishnamoorthy, and D. Sier, "Staff scheduling and rostering: A review of applications, methods and models," *European journal of operational research*, vol. 153, no. 1, pp. 3–27, 2004.

[2] P. M. Berry, M. Gervasio, B. Peintner, and N. Yorke-Smith, "Ptime: Personalized assistance for calendaring," *ACM Transactions on Intelligent Systems and Technology (TIST)*, vol. 2, no. 4, p. 40, 2011.

[3] Microsoft, "Using the calendar in outlook.com," http://windows.microsoft.com/en-us/windows/outlook/calendar-use-calendar, (Accessed on 04/07/2016).

[4] Google, "Google calendar," https://www.google.com/celendar, (Accessed on 04/07/2016).

[5] P. Capek, W. Grey, P. Moskowitz, C. Pickover, and D. Shi, "Event scheduling with optimization," Mar. 11 2008, uS Patent 7,343,312. [Online]. Available: https://www.google.com/patents/US7343312

[6] J. Lofton, "System and method for scheduling events on an internet based calendar," Aug. 14 2003, uS Patent App. 10/037,912. [Online]. Available: https://www.google.com/patents/US20030154116

[7] J. Vincent, "Meeting scheduler with alternative listing," Sep. 17 1991, uS Patent 5,050,077. [Online]. Available: https://www.google.com/patents/US5050077

[8] B. Cragun and P. Day, "Method and meeting scheduler for automated meeting insertion and rescheduling for busy calendars," Oct. 16 2007, uS Patent 7,283,970. [Online]. Available: https://www.google.com/patents/US7283970

[9] D. Conmy, S. Beckhardt, J. Banks-Binici, and R. Slapikoff, "Electronic calendar with group scheduling and storage of user and resource profiles," Jul. 25 2006, uS Patent 7,082,402. [Online]. Available: https://www.google.com/patents/US7082402

[10] D. Conmy and J. Banks-Binici, "Electronic calendar with group scheduling and automated scheduling techniques for coordinating conflicting schedules," Aug. 8 2000, uS Patent 6,101,480. [Online]. Available: https://www.google.com/patents/US6101480

[11] J. Alford, P. Arellanes, J. George, and M. Molander, "Managing flexible events within an electronic calendar," Apr. 20 2010, uS Patent 7,703,048. [Online]. Available: https://www.google.com/patents/US7703048

[12] R. Zhang, O. Brdiczka, V. Bellotti, and J. Shen, "Method and apparatus for automatically managing user activities using contextual information," Mar. 6 2014, uS Patent App. 13/599,750.