Energy-Aware Meeting Scheduling Algorithms for Smart Buildings

Abhinandan Majumdar
Computer Systems Laboratory
Cornell University
abhi@csl.cornell.edu

David H. Albonesi
Computer Systems Laboratory
Cornell University
albonesi@csl.cornell.edu

Pradip Bose
IBM T.J. Watson Research Center
pbose@us.ibm.com

Abstract

The increasing worldwide concern over the energy consumption of commercial buildings calls for new approaches that analyze scheduled occupant activities and proactively take steps to curb building energy use. As one step in this direction, we propose to automate the scheduling of meetings in a way that uses available meeting rooms in an energy efficient manner, while adhering to time conflicts and capacity constraints. We devise a number of scheduling algorithms, ranging from greedy to heuristic approaches, and demonstrate up to a 70% reduction in energy use, with the best algorithms producing schedules whose energy use matches that of a brute force oracle.

Categories and Subject Descriptors

I.2.8 [**Artificial Inteligence**]: Problem Solving, Control Methods, and Search—*scheduling*; G.1.6 [**Numerical Analysis**]: Optimization—*global optimization*

General Terms

Algorithms, Human Factors

Keywords

Smart buildings; Meeting scheduling algorithms; Building energy efficiency

1 Introduction

The energy consumption of commercial buildings is of growing worldwide concern. Buildings constitute 40% of the total U.S. energy consumption, and approximately 74% of the energy consumed by the building sector is derived from fossil fuels [9]. Thus, there is the potential for significant economic and environmental impact by reducing building energy use.

Recent research in the area of HVAC control has focused on making these systems more adaptive to changing conditions. Occupancy sensing and prediction [4, 5, 6, 8, 10, 11, 13, 14] involves the use of cameras, IR motion sensors, optical tripwires, badge readers, or custom presence detectors to track building occupants and react accordingly. For example, HVAC conditioning can be reduced in a currently unoccupied zone, or in one that is predicted to become imminently unoccupied.

As a complement to reactive approaches such as occupancy prediction, we envision future intelligent building systems taking acceptably benign *proactive* measures to affect occupant behavior in a way that improves building energy efficiency. If one considers the occupants as the building workload, reactive techniques such as occupancy prediction take action upon detected changes in the workload–*e.g.*, occupant movements–whereas proactive control approaches attempt to suitably *shape* the workload, to cause the occupants to take relatively benign actions that are more energy-friendly.

One way to shape the building occupant workload to save energy is through the scheduling of meetings to available meeting rooms. Currently, meeting scheduling is largely ad hoc with at best some attention paid to meeting room capacity. Given a complex schedule of meeting times and occupied rooms, there are many possible schedules and great differences in their energy usage. The determination of a reasonably energy-efficient schedule is a non-trivial exercise due to the many factors that impact the energy use of a meeting schedule, including the number of occupied rooms, the time gap between successive meetings in the same room, and the match of room size to meeting size. To address this problem, we envision future workplaces where meeting locations are determined by an energy-aware smart meeting scheduler and automatically inserted into electronic calendars.

In this paper, we propose and evaluate a wide range of smart meeting scheduling algorithms. Given a set of scheduled meeting times and a set of meeting rooms in relatively close proximity, the scheduler spatially assigns meetings to rooms in a way that avoids time conflicts and respects meeting capacity constraints while minimizing meeting room energy consumption. We create a variety of algorithms based on backtracking, greedy, and heuristic approaches, and perform a detailed evaluation for a number of meeting schedules and different climates. Our results demonstrate up to a 70% energy savings. Moreover, the heuristic algorithms perform competitively against an oracle exhaustive search algorithm while significantly reducing computational complexity.

Buildsys'12, November 6, 2012, Toronto, ON, Canada.
Copyright © 2012 ACM 978-1-4503-1170-0 ...$10.00

2 Meeting Scheduling Problem Description

Given a set of meetings and available rooms, the overall objective is to allocate rooms to the meetings in such a way that the HVAC energy in conditioning the rooms is minimized while maintaining the desired set temperature.

Let R and M be the set of n rooms and m meetings, respectively. For each i^{th} meeting $M_i \in M$, $M_i.st$, $M_i.et$, $M_i.size$, and $M_i.room$ represent the start time, end time, meeting size and the room where M_i is scheduled, respectively. Initially, $M_i.room = \phi$. For each j^{th} room $R_j \in R$, $R_j.capacity$ denotes its capacity.

The room scheduling problem is formulated as an optimization problem. The objective is to determine a meeting schedule that minimizes HVAC energy for all the rooms while maintaining thermal comfort (set temperature objectives). The scheduled meetings must be free from timing conflicts (meetings do not overlap within the same room) and capacity mismatches (a scheduled meeting does not exceed the room capacity).

The objective function is:

$$\texttt{Minimize} \sum_{j=1}^{n} Energy(R_j) \tag{1}$$

with the following constraints:

- Timing

$$M_i.st < M_i.et \leq M_k.st < M_k.et \tag{2}$$

$$\forall j \text{ and } \forall i \neq k \text{ with } M_i.room = M_k.room = R_j$$

- Capacity

$$M_i.size \leq R_j.capacity \tag{3}$$

$$\forall\, M_i.room = R_j$$

2.1 Factors Impacting HVAC Energy of a Meeting Schedule

Given a particular building design, HVAC system, and climate, the following factors affect the HVAC energy of a meeting schedule:

Per Room Usage: The length of time that a room is occupied. A room occupied for a longer time consumes more energy than a shorter time. Moreover, a larger room requires more energy than a smaller room when occupied for the same amount of time.

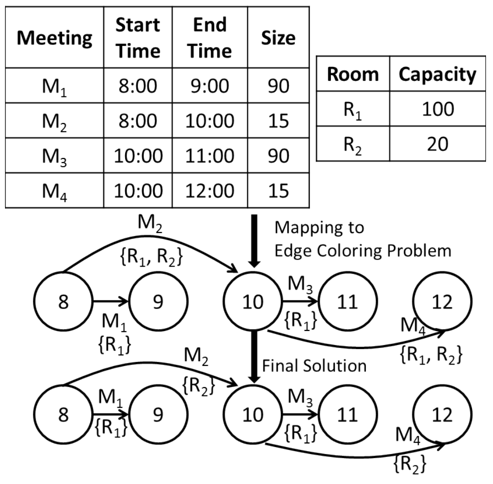
Capacity Size Difference: The difference between the room capacity and the meeting size. A room that is sized appropriately is more energy-efficient than a room that is oversized.

Time Gap: The time interval between meetings when a room is unoccupied. When a meeting ends, the room is pre-conditioned to a comfortable environment. This built-up thermal momentum can benefit later meetings scheduled in the same room in close proximity.

Number of Occupied Rooms: Assuming that conditioning of unoccupied rooms can be reduced to save energy, more occupied rooms require more energy than fewer ones.

3 Meeting Scheduling Algorithms

The scheduling of m meetings in n rooms is similar to an edge coloring problem [7]. Each node of the graph represents a specific time and a directed edge refers to a meeting, with the start and end nodes corresponding to the start and end meeting times, respectively. An example graph is shown in Figure 1. If each room is represented by a different color, an edge (meeting) can be assigned a specific color (room) such that both constraints are met. Specifically, the `Timing` constraint requires that no two edges that overlap in time be assigned the same color, and the `Capacity` constraint demands that edges be marked with only an allowed set of colors. In Figure 1, all the edges (meetings) are associated with a set of allowed colors (rooms), and there are no time overlapping edges (meetings) of the same color (room).



Meeting	Start Time	End Time	Size
M_1	8:00	9:00	90
M_2	8:00	10:00	15
M_3	10:00	11:00	90
M_4	10:00	12:00	15

Room	Capacity
R_1	100
R_2	20

Figure 1. Example showing meeting scheduling as an edge coloring problem

The edge coloring problem is, in general, NP-complete and for our meeting scheduling problem the number of possible solutions without timing conflicts or capacity mismatches is n^m. For instance, for ten non-conflicting meetings that can be scheduled in any four rooms, there are over a million possible solutions. Thus, we propose a number of alternative meeting scheduling algorithms with more modest complexity. These algorithms can be categorized into *backtracking*, *greedy*, and *heuristic* approaches.

The backtracking and greedy algorithms use meeting times, the number of attendees, and room capacities to build a meeting schedule based on particular criteria, *e.g.*, minimizing the number of occupied rooms or minimizing the time gaps between meetings in the same room. The heuristic-based algorithms are guided by an analytical model and yield consistently better results across a range of meeting scenarios. Our results show that this heuristic approach combined with selected EnergyPlus simulations and backtracking finds solutions that are within 1% of brute force exhaustive search while requiring orders of magnitude fewer simulations.

3.1 Backtracking Algorithms

Brute Force (`BF`): `BF` exhaustively searches all possible meeting room combinations within the constraint boundaries and returns the minimum energy solution as determined by EnergyPlus. We use this algorithm as an oracle against which we compare the performance of the other algorithms.

`BF` starts with the first meeting and recursively generates a search tree from the meeting graph, thereby exploring all possible room choices. For a meeting M_i, all feasible room options as per the constraints are tested and assigned iteratively. For every such assignment made, the algorithm recursively schedules the $(i+1)^{th}$ meeting and so on. In this way, `BF` performs a depth-first search of all the leaf nodes ($i = m$), each of which represents one unique solution. At $i = m$, EnergyPlus is invoked to return the energy of this schedule.

The complexity of `BF` when there is no time conflict or capacity mismatch is $O(n^m)$. Although conflicts and mismatches may reduce the search space in some cases, each solution requires an EnergyPlus simulation, which makes `BF` computationally impractical. For instance, EnergyPlus simulations of the 1,048,576 solutions for scheduling 10 continuous non-overlapping meetings in 4 rooms requires 42 days of execution time using a 2.7GHz Intel® Core i7-2620M Processor with 8GB of DRAM.

Random Room: We use `Random Room` as a practical baseline algorithm, while `BF` serves as an impractical oracle. `Random Room` is similar to `BF`, with a difference that it randomly performs the depth search (unlike `BF` which searches all room options) and stops after it finds its first solution. A meeting M_i is randomly assigned to room R_j while meeting the constraints, and the recursion continues with the $(i+1)^{th}$ meeting. If at any stage, there is no such possible room options because of either constraint, the algorithm backtracks to its parent function. The parent function then retries with the next randomly selected but different room option and again continues the recursion, until all the meetings are scheduled.

Maximum/Minimum Room: These algorithms are similar to `Random Room`, except that they prioritize the room selection based on the duration that a room is kept occupied. In each step, `Maximum Room` selects the room that is occupied for the least amount of time in order to use as many rooms as possible, while `Minimum Room` selects the room that is occupied for the most amount of time. To prioritize room selection, the algorithms maintain a priority queue of room usage. Initially, the room priorities are randomly generated to ensure fair scheduling. As the algorithm progresses, the priorities are reevaluated each time a meeting is scheduled. `Maximum Room` attempts to equalize the meeting allocations, and thus ends up using maximum rooms. `Minimum Room` reuses a room already allotted to a particular meeting.

3.2 Non-Heuristic Greedy Algorithms

The greedy algorithms described in this section follow a particular strategy that yields an optimum solution for some cases. In other cases, they may produce a sub-optimal solutions or even a worse solution than `Random`, or may be unable to find a solution altogether. However, the algorithms have polynomial time complexity and do not rely on EnergyPlus simulations.

Minimum Gap (Longest/Shortest Continuous): These algorithms attempt to reduce the periods of inactivity (time gaps) between back-to-back meetings in order to take advantage of thermal momentum. For these algorithms, a meeting M_i is first randomly scheduled in a room R_j such that it doesn't violate the constraints. Next, all adjacent meetings with start or end times equal to that of M_i are scheduled in room R_j while meeting the constraints. Selection of M_i and resolution of the case of multiple adjacent meetings leads to two strategies. `Longest Continuous` selects the longest meeting first, and therefore keeps the room busy for a longer duration. However, this approach may create an imbalance due to shorter discontinuous meetings spread out over other rooms. If longer meetings are prioritized first, then the leftover smaller meetings may create gaps and may require exclusive allocation to other rooms. `Shortest Continuous` balances the meeting distribution by prioritizing smaller meetings. The complexity of the algorithms is $O(m^2n + m^3)$

Minimum Capacity Difference: In this algorithm, the i^{th} meeting M_i is scheduled to a room R_j such that the difference ($R_j.capacity$ - $M_i.size$) is minimized, while meeting the constraints. Since conditioning and maintaining a bigger room to a desired set temperature consumes more HVAC energy than conditioning a smaller room, the idea is to avoid scheduling a meeting with few attendees in a large room. The meetings are scheduled in order of their room choices based on the `Capacity` constraint, from fewest choices to most choices. The complexity of this algorithm is $O(m^2n)$.

Minimum Capacity Difference + Minimum Gap: This algorithm combines `Minimum Capacity Difference` and `Minimum Gap`. The meetings are scheduled in order of their room choices based on the `Capacity` constraint, from fewest choices to most choices. An initial room assignment is made for meeting M_i using `Minimum Capacity Difference`. Then all adjacent meetings of M_i (and their adjacent meetings) are scheduled in the same room where M_i was scheduled. When multiple adjacent meetings for $i \neq k$ are present, the one that has minimum capacity difference is selected. The `Minimum Capacity Difference` algorithm matches meetings as per room capacity, but creates gaps and ends up using more rooms. The present algorithm schedules discontinuous meetings to the rooms as per its capacity and allocates adjacent meetings to the same room. In this way, the algorithm attempts to avoid gaps and uses fewer rooms to schedule meetings, but creates capacity mismatches for the adjacent meetings. The complexity of the algorithm is $O(m^2n + m^3)$.

We also developed other combined algorithms, such as `Minimum Gap (Longest/Shortest Continuous) + Minimum Room`, but they were less effective overall than `Minimum Capacity Difference + Minimum Gap`.

3.3 Heuristic Algorithms

The heuristic search algorithms that we developed use an analytical model (discussed in Section 3.3.1) of carefully selected factors to approximate the energy-efficiency of a given meeting schedule. We use an analytical model due to its computational speed. Runtimes would be prohibitive if EnergyPlus simulations were required at each step of the algorithms.

Hybrid Greedy: `Hybrid Greedy` is a heuristic-guided greedy algorithm. As with `Minimum Capacity Difference`, meetings are scheduled in the order of their room choices based on the `Capacity` constraint, from fewest choices to most choices. When multiple meetings have an equal number of room choices, meetings are processed in the order of increasing start time (earlier meetings are handled first).

For the non-conflicting situation with only one meeting between any two time stamps, `Hybrid Greedy` explores all the room choices that meet the constraints and compares them using the analytical model. The meeting is scheduled in the room with the lowest estimated cost as per the heuristic.

There are two cases to consider when multiple meetings start at the same time. In the first case, the conflicting meetings have different room choices that minimize their cost, in which case each conflicting meeting is scheduled in its lowest-cost room. In the second case, multiple meetings have the same lowest-cost room. For *n* conflicting meetings, finding a globally optimum schedule in this case is O(n!).

We developed an effective greedy approach to this situation in order to reduce the computational complexity to polynomial time. For each conflicting meeting, we determine the cost difference between the lowest cost and second lowest cost room choices. The meeting with the highest cost difference is given the lowest cost room. For the remaining meetings, we iterate with the second lowest cost option, and continue until all conflicting meetings are assigned a room. While this approach is not guaranteed to find the optimum solution, we found that it performed comparably to `BF` over a variety of benchmarks.

A** Search: `A`* `Search` is a heuristic-based backtracking algorithm that reduces the search path traversal of `BF` by avoiding those depth searches whose sub-optimality can be determined *a priori*. It uses the analytical model along with selected EnergyPlus simulations in traversing the graph. During the depth-first search of various room options for meeting M_i in the i^{th} recursion, the algorithm evaluates the estimate of the optimality of the generated schedule through the heuristic (without invoking the EnergyPlus simulator). The *(i+1)*th recursive step is called only if the returned value from the analytical model is smaller than that of the current optimal schedule. Otherwise, the search tree is pruned at this point. Whenever the algorithm reaches a leaf node ($i = m$), the algorithm invokes EnergyPlus to calculate the energy consumption of this schedule. If the returned energy is smaller than the current optimum, the optimum energy value, schedule and corresponding optimum heuristic value are updated.

3.3.1 Analytical Model

`Hybrid Greedy` uses a heuristic in its cost analysis. `A`* `Search` uses the same heuristic to guide the search tree traversal and avoid searching sub-optimal paths. The heuristic is based on the following factors that approximate the energy-optimality of a meeting schedule.

***Per Room Usage* (`usg_val`)**: This factor accounts for the per-room usage of the scheduled meetings. In general, conditioning a larger room consumes more HVAC energy than a smaller room. Similarly, scheduling a longer meeting in a larger room is energy-inefficient compared to scheduling it in a smaller room. `usg_val` captures the tradeoff between a short meeting scheduled in a large room versus a small room allotted to a long meeting. Mathematically, for each room R_j, `usg_val`$_j$ is the ratio of the total time R_j is occupied with all the scheduled meetings and the maximum duration R_j can possibly be occupied:

$$usg\_val_j = \frac{\sum_{i=1}^{m}(M_i.et - M_i.st)}{\max_i(M_i.et) - \min_i(M_i.st)} \quad (4)$$

$$\forall\ 1 \leq i \leq m,\ 1 \leq j \leq n \text{ and } M_i.room = R_j$$

The value of `usg_val`$_j$ ranges from 0 to 1, and a smaller value indicates lower energy use. Finally, `usg_val` is the average of all `usg_val`$_j$ weighted as per individual room capacities of room R_j:

$$usg\_val = \frac{\sum_{j=1}^{n}(R_j.capacity \times usg\_val_j)}{\sum_{j=1}^{n} R_j.capacity} / n \quad (5)$$

***Per Room Size Difference* (`size_val`)**: This factor estimates the per-room difference in the room capacity and meeting size. Scheduling a small meeting in a large room is less efficient than scheduling it in a smaller room. Mathematically, `size_val`$_j$ for a room R_j is the ratio of the sum of the difference between R_j's capacity and the meeting size, and the capacity difference for the smallest meeting:

$$size\_val_j = \frac{\sum_{i=1}^{m}(R_j.capacity - M_i.size)}{R_j.capacity - \min_i(M_i.size)} \quad (6)$$

$$\forall\ 1 \leq i \leq m,\ 1 \leq j \leq n \text{ and } M_i.room = R_j$$

The value of `size_val`$_j$ ranges from 0 to 1, and a smaller value indicates lower energy use. Finally, `size_val` is the average of all `size_val`$_j$ weighted as per individual room capacities of room R_j:

$$size\_val = \frac{\sum_{j=1}^{n}(R_j.capacity \times size\_val_j)}{\sum_{j=1}^{n} R_j.capacity} / n \quad (7)$$

***Per Room Gap* (`gap_val`)**: This factor represents the per-room time gap when a room is unoccupied and no meeting is scheduled. Generally, when there is no gap between subsequent meetings, energy optimality is the highest. A large enough time gap permits the HVAC system to reduce the room to its minimum temperature, thereby saving energy. Shorter time gaps between meetings cause inefficiencies. `Gap_val` models this tradeoff. Mathematically, `gap_val`$_j$ is 0 when room R_j has no time gap between successive scheduled meetings. Otherwise, it is one minus the ratio of the time gap between all scheduled meetings in R_j over the maximum time gap possible. Assuming the shortest meeting is of one hour duration, the denominator of equation 8 corresponds to the maximum sized gap between two one hour meetings, one at the beginning of the day and the other at the end of the day:

$$gap\_val_j = \begin{cases} 0 & \text{if } M_i.et = M_k.st \\ 1 - \frac{\sum_{i,k=1}^{m}(M_k.st - M_i.et)}{\max_i(M_i.et) - \min_i(M_i.st) - 2} & \text{if } M_i.et \neq M_k.st \end{cases} \quad (8)$$

$$\forall\ 1 \leq i < k \leq m,\ 1 \leq j \leq n \text{ and } M_i.room = M_k.room = R_j$$

The value of `gap_val`$_j$ ranges from 0 to 1, and a smaller value indicates lower energy use. Finally, `gap_val` is the average of all `size_val`$_j$ weighted as per individual room capacities of room R_j:

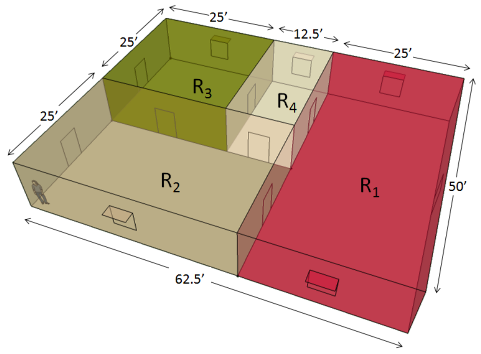


Figure 2. Layout of the simulated building

Table 1. Room Area and Capacities

Rooms	Area (in ft^2)	Capacity
R_1	1250	104
R_2	937.5	78
R_3	625	52
R_4	312.5	26

$$gap\_val = \frac{\sum_{j=1}^{n}(R_j.capacity \times gap\_val_j)}{\sum_{j=1}^{n} R_j.capacity}/n \qquad (9)$$

***Num Rooms* `(num_rooms)`**: This factor captures the number of rooms out of n rooms used to schedule m meetings. More rooms require more HVAC energy.

The value of the heuristic is calculated as the product of `num_rooms` and the average of `usg_val`, `size_val`, and `gap_val`, with each of the these factors appropriately weighted. For our building layout and meeting benchmarks, we determined that equal weighting of the factors produced results for `Hybrid Greedy` and `A* Search` that were very close to that of `BF`. We leave as future work the investigation of the variability of the relative importance of these factors with different building characteristics and external conditions.

4 Experimental Setup

The layout of the simulated building is shown in Figure 2. The building contains four rooms of different sizes and capacities as shown in Table 1. We chose to eliminate other spaces (offices, hallways, *etc.*) from the layout in order to focus on meeting room energy savings. We compute room capacities by assuming 12 square feet of area per person for a theatre style room [12]. For energy modeling, we use the Department of Energy building energy simulation software EnergyPlus version 7.0 [1]. The layout is designed using the Google Sketchup Tool [3] with the OpenStudio plugin to support whole building energy modeling using EnergyPlus [2]. The construction material for building surface and fenestration, and schedules for lighting and electrical equipment, are exported from an existing large office template. Each room maps to a unique thermal zone, and is controlled by an individual zone-level thermostat. The HVAC control system uses the default Ideal Air Loads System.

We simulate three locations: San Diego, Phoenix, and Minneapolis, each over a five day period. Phoenix and Minnesota are simulated for the peak summer (July 14-18) and winter (January 28 to February 1) weather, respectively. San Diego is simulated from June 5-9 when the outside temperature is close to the indoor set temperature. During cooling season, the set temperature for an unoccupied room is 26.7°C, while that for an occupied room is 24°C. For the heating season, the set temperature for an unoccupied room is 15.6°C and that for an occupied room is 21°C. Room level thermostats are set to the occupied set temperature 15 minutes prior to the start time of a scheduled meeting, which we experimentally verified is sufficient time to condition the room before a meeting begins in all three climates. Similarly, when there is no subsequent meeting, thermostats are set to the unoccupied temperature 15 minutes after the current meeting ends.

The meeting scheduling algorithms are implemented in C. A function `EnergyPlus(`*M*`)` invokes a Perl script that calculates the building energy for an input meeting schedule. The Perl script takes the building configuration (`.idf`) file as an input, maps the rooms from the schedule to the thermal zones of the building, creates the thermostat and occupancy schedules and invokes the EnergyPlus simulator through a batch script. A parser processes the EnergyPlus result files and calculates the cumulative HVAC energy of the entire building. Measurements begin an hour before the first scheduled meeting and end an hour after the last one to ensure that the rooms are preconditioned to a minimum comfortable temperature and to reduce the effect of external factors on the HVAC energy measurements.

The algorithms are evaluated using a variety of meeting benchmarks (Figure 3). The first five cases are synthetically designed to test special cases. The last two benchmarks are randomly generated meeting schedules. The benchmarks are represented by graphs, with nodes representing time stamps, edges indicating meetings, and the number in brackets indicating the meeting size. Meetings are scheduled between 8am and 6pm with a minimum duration of one hour.

5 Results

The energy of the meeting scheduling algorithms relative to the baseline `Random Room` algorithm for the seven benchmarks is shown in Figure 4 for San Diego. The percentage energy savings is shown in the top graph while absolute energy savings is shown below.

We first consider the algorithms that use a single criterion in their decision making. For most of these algorithms, the use of a single criterion yields poor results, in some cases even worse than random scheduling. For the synthetic benchmarks with a serial meeting schedule, `Maximum Room` is far from the best choice, since in attempting to spread the meetings out over many rooms it fails to take advantage of thermal momentum. `Minimum Room` only fares slightly better despite assigning all meetings to the same room. This is because it selects a room at random from among all those that meet the `Capacity` constraint, and may choose a room that is larger than necessary. Both of these algorithms perform better for other cases, but are never close to the best.

Between the two `Minimum Gap` strategies, `Shortest Continuous` performs better for `9m_15_90` since in this case prioritizing shorter meetings creates fewer gaps than priori-

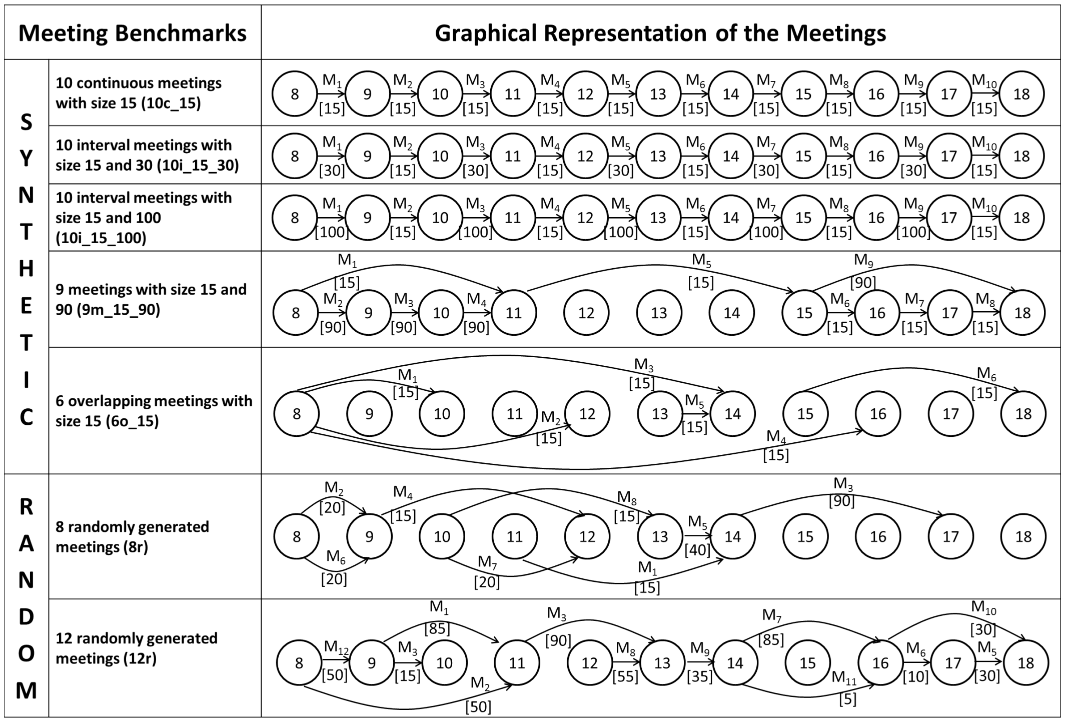

Meeting Benchmarks
Graphical Representation of the Meetings
SYNTHETIC
10 continuous meetings with size 15 (10c_15)
10 interval meetings with size 15 and 30 (10i_15_30)
10 interval meetings with size 15 and 100 (10i_15_100)
9 meetings with size 15 and 90 (9m_15_90)
6 overlapping meetings with size 15 (6o_15)
RANDOM
8 randomly generated meetings (8r)
12 randomly generated meetings (12r)

Figure 3. Meeting benchmarks

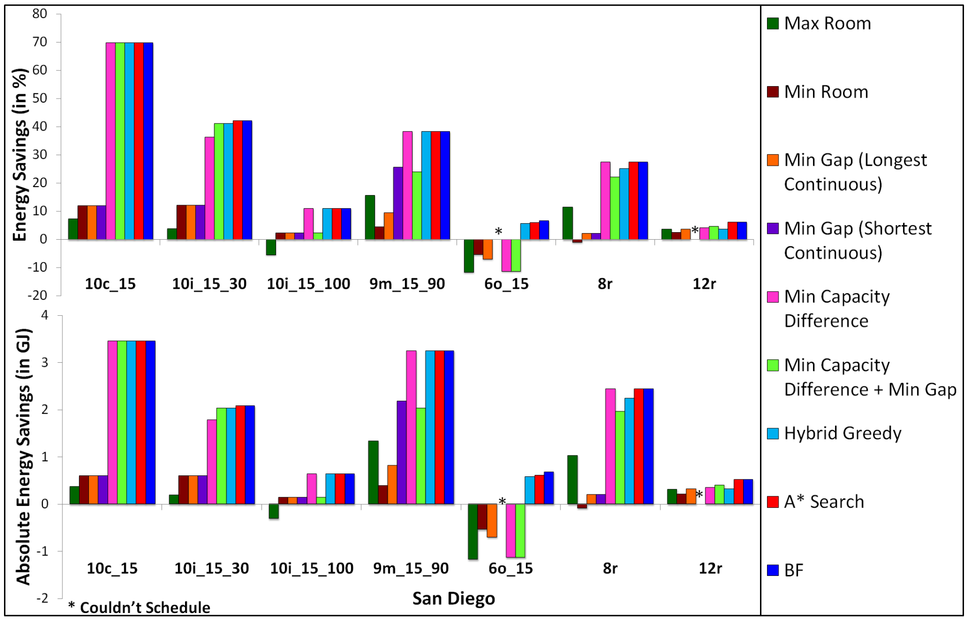
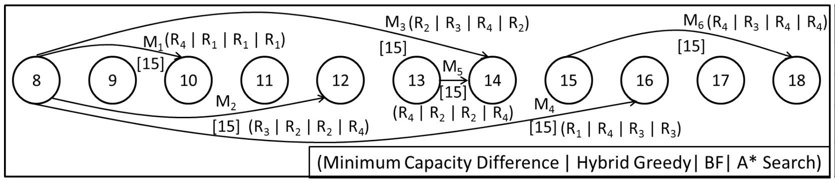


Figure 4. HVAC energy savings over `Random Room` for San Diego

tizing long ones. However, the algorithm falls short of the best algorithms due to its failure to account for capacity differences. Moreover, `Shortest Continuous` is unable to find a solution for the `6o_15` and `12r` test cases.

Among the single criterion approaches, `Minimum Capacity Difference` is by far the best, due to the fact that minimizing the difference between room capacity and the number of meeting attendees is the single biggest factor in reducing energy use. For two of the three serial synthetic benchmarks (`10c_15` and `10i_15_100`), `Minimum Capacity Difference` matches the oracle `BF` algorithm. Surprisingly though, it falls short for the `10i_15_30` serial case. Here, the difference between the room capacities is small enough that it is better to schedule all the meeting in the same room (R_3) to eliminate gaps (and not lose thermal momentum) and use fewer rooms than to minimize capacity differences (alternately schedule in R_3 and R_4). `Minimum Capacity Difference + Minimum Gap` does the former and therefore



Room	Capacity
R_1	104
R_2	78
R_3	52
R_4	26

Figure 5. Schedule comparison for `6o_15` benchmark

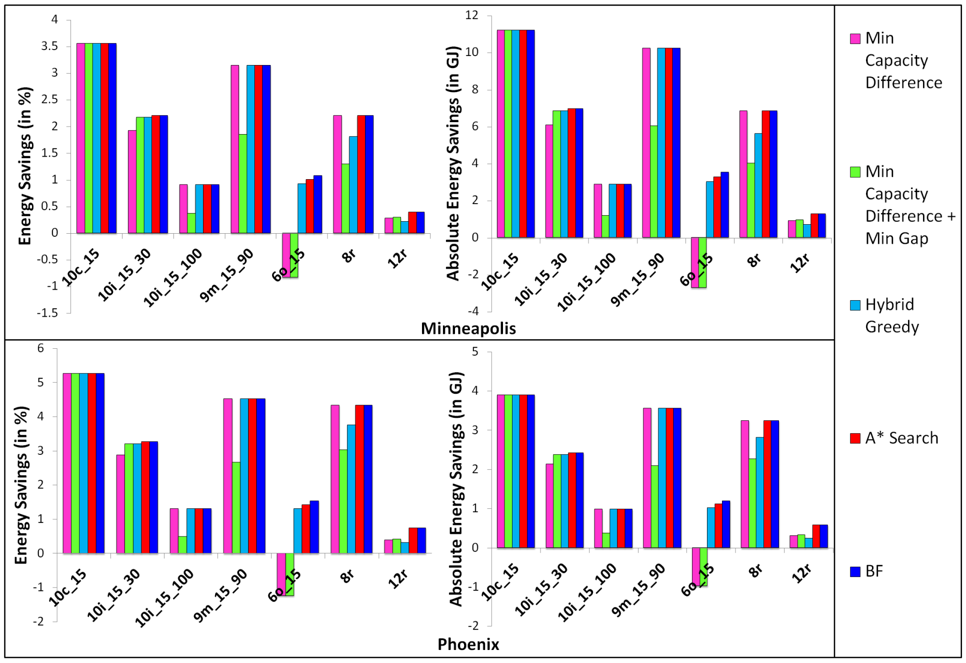


Figure 6. HVAC energy savings over `Random Room` for Minneapolis and Phoenix

performs better in this case, though no better in the others.

Both `Minimum Capacity Difference` and `Minimum Capacity Difference + Minimum Gap` perform poorly for `6o_15`. One of the challenges of this schedule is that there are conflicting meetings with the same capacity starting at the same time of 8:00. Because the capacity differences are identical, neither algorithm (nor any other combined algorithm) is able to come close to the optimal schedule, and they even perform worse than random scheduling for this case.

The `Hybrid Greedy` and `A* Search` heuristic algorithms perform consistently close to optimal across all benchmarks, in part due to the analytical model to guide their decision-making. `Hybrid Greedy` properly schedules all meetings in the same room for `10i_15_30`, and more intelligently schedules `6o_15` where multiple conflicting meetings start at the same time, demanding the same room as their lowest cost option. However, since `Hybrid Greedy` is greedy and non-backtracking, it is sometimes outperformed by `A* Search`, which performs identically to `BF` except for `6o_15`, where it differs by only 0.7%. As shown in Table 2, `A* Search` achieves this performance while requiring orders of magnitude fewer EnergyPlus simulations than `BF`. Figure 5 compares the room assignments made by the `Minimum Capacity Difference`, `Hybrid Greedy`, `BF`, and `A*` algorithms for the `6o_15` case. `Minimum Capacity Difference` cannot distinguish among the choices at 8:00 and makes a far-from-optimal decision. `Hybrid Greedy` makes the optimal greedy decision by scheduling the longest meeting in the smallest room, the next longest meeting in the second smallest room, and so on. At 13:00, it picks the smallest available room, and again at 15:00. Interestingly, the choices made by the oracle `BF` algorithm are subtly different. Here, the optimal strategy is to assign the smallest room (R_4) to M_3 so that it can be reassigned to M_6 at 15:00. For this particular scenario, the most energy efficient choice is to look ahead and find the longest non-overlapping path and assign to it the smallest room, rather than the greedy strategy employed by `Hybrid Greedy`. While `A* Search` makes a slightly less-optimal decision than `BF`, it requires 16X fewer EnergyPlus simulations.

The energy savings for Minneapolis and Phoenix for the most effective algorithms is shown in Figure 6. The performance trends of the algorithms for the different benchmarks are similar as for San Diego. However, a significant amount of energy is spent in conditioning unoccupied rooms due to the extreme outdoor air temperatures in these locations. Thus, the percentage energy savings for Minneapolis and Phoenix are much smaller than for San Diego. Specifically, the `BF` algorithm yields 7-70% energy savings in San Diego, but 0.7-3.6% for Minneapolis and for Phoenix 1-5.3%. However, these more extreme conditions cause the absolute energy savings for Minneapolis and Phoenix to be

Table 2. EnergyPlus simulations required by `BF` and `A`* `Search`

Benchmarks	BF	A* Search
`10c_15`	1,048,576[1]	1
`10i_15_30`	248,832[1]	12
`10c_15_100`	1024	31
`9m_15_90`	324	1
`6o_15`	144	9
`8r`	1440	6
`12r`	324	42

several times higher than for San Diego. Thus, assuming equal energy costs among regions, smart meeting scheduling yields higher percentage savings in more temperate climates, but higher absolute cost savings in more extreme climates.

Overall, our work yields the following insights:

- In general, algorithms based on one or two criteria produce far-from-optimal results, and may perform far worse than random scheduling for some cases.
- Capacity matching is the single most important individual criterion, and a capacity matching algorithm can perform well for a number of simple scheduling scenarios. However, for others it can perform far worse than optimal, and worse than random.
- A carefully constructed heuristic can significantly improve algorithm performance and consistency across different meeting scenarios, even for greedy algorithms that have the advantage of operating in polynomial time.
- A heuristic-guided `A* Search` algorithm can match the performance of exhaustive EnergyPlus simulations while requiring orders of magnitude fewer simulations.
- Due to the high energy cost of conditioning unused rooms, the percentage energy improvement is lower at more extreme climates (Phoenix and Minneapolis) than more temperate ones (San Diego). However, the absolute savings is more significant for extreme climates.

6 Related Work

The work by Pan et. al [15, 16] is the only prior research to our knowledge that proposes intelligent meeting room management to reduce building energy use. The authors develop a sensor network whose purpose is to build an energy-temperature correlation model and two scheduling algorithms for non-uniform meeting rooms. The first greedy algorithm that assigns the meetings to the rooms with minimum capacity difference in an increasing order of start time. The second reschedules the meetings in time. Their first algorithm is closest to our `Minimum Capacity Difference` algorithm. They do consider reducing the time gap but only for those meetings that have minimum capacity difference. Our work considers other factors such as per room usage and total number of rooms; handles the situation where meetings starting at the same time request the same minimum cost room; and explores combinations of various factors and a heuristic-guided search approach that reduces the search space (EnergyPlus simulations) of brute force exhaustive search. Furthermore, we also show that our schemes perform consistently across a wide range of benchmarks.

7 Conclusions

The energy consumption of commercial buildings is of growing worldwide concern, which calls for new approaches that analyze scheduled building occupant activities and proactively take steps to curb building energy use. We propose smart meeting scheduling algorithms that use available meeting rooms in a more energy efficient manner, while adhering to time conflicts and capacity constraints. We devise a number of scheduling algorithms, ranging from greedy to heuristic approaches, and through a detailed evaluation of a range of benchmarks and multiple climates, demonstrate up to a 70% reductions in energy use. Moreover, the best algorithms produce schedules whose energy use matches that of a brute force oracle approach.

8 References

[1] EnergyPlus Energy Simulation Software. http://apps1.eere.energy.gov/buildings/energyplus/.

[2] NREL: OpenStudio. http://openstudio.nrel.gov.

[3] Trimble Sketchup. http://sketchup.google.com.

[4] Y. Agarwal, B. Balaji, R. Gupta, J. Lyles, M. Wei, and T. Weng. Occupancy-Driven Energy Management for Smart Building Automation. In *Proceedings of the 2nd ACM Workshop on Embedded Sensing Systems for Energy-Efficiency in Buildings*, pages 1–6, 2010.

[5] R. H. Dodiera, G. P. Henzeb, D. K. Tillerb, and X. Guob. Building Occupancy Detection through Sensor Belief Networks. *Energy and Buildings*, 38(9):1033–1043, 2006.

[6] V. L. Erickson, Y. Lin, A. Kamthe, R. Brahme, A. Surana, A. E. Cerpa, M. D. Sohn, and S. Narayanan. Energy Efficient Building Environment Control Strategies Using Real-time Occupancy Measurements. In *Proceedings of the First ACM Workshop on Embedded Sensing Systems for Energy-Efficiency in Buildings*, pages 19–24, 2009.

[7] T. R. Jensen and B. Toft. *Graph Coloring Problems*, chapter 12, pages 190–203. Wiley-Interscience, Dec. 1994.

[8] X. Jiang, B. Dong, and L. Sweeney. From Sensor Network To Social Network- A Study On The Energy Impact In Buildings. In *NIPS Workshop on Analyzing Networks and Learning with Graphs*, pages 1–7, Dec. 2009.

[9] J. D. Kelso. *2011 Buildings Energy Data Book*, chapter 1. D&R International, Ltd., Mar. 2012.

[10] J. Krumm and A. J. B. Brush. Learning Time-Based Presence Probabilities. In *9th International Conference on Pervasive Computing*, pages 79–96, 2011.

[11] C. Liao, Y. Lin, and P. Barooah. Agent-based and Graphical Modeling of Building Occupancy. *Journal of Building Performance Simulation*, 5:5–25, Jan. 2012.

[12] D. Lutz. *Guide To Meeting Management*, page 63. Convene, Sept. 2000.

[13] S. Meyn, A. Surana, Y. Lin, S. Oggianu, S. Narayanan, and T. Frewen. A Sensor-Utility-Network Method for Estimation of Occupancy in Buildings. In *Proceedings of the 48th IEEE Conference on Decision and Control*, pages 1494 –1500, Dec. 2009.

[14] J. Page, D. Robinson, N. Morel, and J.-L. Scartezzini. A Generalised Stochastic Model for the Simulation of Occupant Presence. *Energy and Buildings*, 40(2):83 – 98, 2008.

[15] D. Pan, Y. Yuan, D. Wang, X. Xu, Y. Peng, X. Peng, and P.-J. Wan. Thermal Inertia: Towards An Energy Conservation Room Management System. In *INFOCOM*, pages 2606 –2610, Mar. 2012.

[16] D. Pan, Y. Yuan, D. Wang, X. Xu, Y. Peng, X. Peng, and P.-J. Wan. Thermal Inertia: Towards An Energy Conservation Room Management System. Technical report, 2012.

[1]By inspection of the benchmark, we were able to find the optimal without simulating this many cases.