Designing adaptive lighting control algorithms for smart buildings and homes

Conference Paper · April 2014

DOI: 10.1109/ICNSC.2014.6819639

2 authors, including:

Yuan Wang
Arizona State University
3 PUBLICATIONS **4** CITATIONS

SEE PROFILE

Designing Adaptive Lighting Control Algorithms for Smart Buildings and Homes

Yuan Wang
Arizona State University
Tempe, AZ, USA
Email: Yuan.Wang.4@asu.edu

Partha Dasgupta
Arizona State University
Tempe, AZ, USA
Email: partha@asu.edu

***Abstract*—Artificial lighting is often the main lighting provision for workplaces. This paper describes algorithms for optimizing lighting control in large (smart) buildings that are extensible to smart home use. Systems that provide uniform lighting, under varying outdoor light levels, at occupied locations turns out to be a hard problem. We present methods that work as generic control algorithms and are not preprogrammed for a particular building. Our system uses wireless sensors and wired actuators for the lighting control. But the system does not know about locations and correlations between lights and sensors. The model shows that the control problem is NP-Hard. A heuristic algorithm is proposed and validated to solve the problem to compute approximate optimal solution.**

I. Introduction

Artificial lighting is essential in large buildings and homes. Lighting has very important effect on people's health and productivity as shown by recent studies [1]. Controlling lighting based on occupancy, daylight effects and energy costs can impact energy usage.

Currently there are 1.5 million commercial buildings and 114 million homes in the US with a growth rate of about 3% each year. Smart buildings and homes need, among other features, the ability to control lighting. Deploying such systems in existing buildings is most feasible if the system uses wireless sensors, simple actuators and does not need custom programming. These goals have driven our research.

Automating the lighting control systems such that uniform lighting, under varying outdoor light levels, is maintained at occupied locations turns out to be a hard problem. We are working on algorithms that can be used in generic systems that are not preprogrammed for a particular building. That is the system does not know about locations and correlations between lights and sensors.

Several existing approaches are given to increase lighting comfort levels. Some of lighting control systems may be able to provide enough lighting for a workplace, but people may still feel uncomfortable when illumination level is sufficient but not uniform. According to [1], three problems are insufficient light or uneven light or lights too bright. Controlling lighting levels automatically need customized control systems that rely on extensive pre-programming involving detailed custom models and/or lengthy set-up. When the pattern in a workplace changes, it needs expensive customization and updates performed by trained personnel, which is costly and time-consuming. The "sequential lighting changes" approach is sometimes used in lighting control system [2] for calculating light settings. This method adjusts light settings based on feedback data, which is able to generate a reasonable result, but the average computational time is high due to the steps of prediction and adjusting. It also increases the number of times lights are switched on/off (unnecessarily) due to its feedback design [2].

In this paper, a Wireless Sensor Network(WSN)-based lighting control system is introduced to efficiently and adaptively control artificial lights to provide a stable and uniform lighting environment. The system measures light at preset locations and turns on or off light switches to achieve the goal. The system is not pre-programmed and learns about effects light switches have on sensors via calibration. The work is formalized using non-linear integer programming model that proves to be NP-Hard. A heuristic algorithm is proposed and validated to solve the problem to compute approximate optimal solution. The approach can be used in different places where uniform lighting environment is required.

For the sake of brevity and simplicity we only describe the operation of the system while there is no outside lighting. Addition of varying levels of outside lights is an extension(see section V). Also, energy consumption minimization can be incorporated without major changes.

The rest of this paper is organized as follows. Section II introduces the background and related work of the problem. Section III presents the heuristic algorithms for lighting control. Experimental work and simulation results are discussed in section IV. Section V talks about potential extensions of the algorithm. We conclude the paper in section VI.

II. Background and Related Work

A. Problem Description

Assume a place has n light switches and m light-level sensors, placed by the human designer of the place. The sensors are connected to the control system via a WSN and the switches are activated via actuators connected to the system. The physical location of lights, sensors and their correlations are initially unknown to the control system.

The ultimate task is to compute the positions (on/off) of n light switches. Let $x = \langle x_1, ..., x_n \rangle$ denote the assignment for light switches where $x_i \in \{0, 1\}$ and 0 denotes off and 1

978-1-4799-3106-4/14/$31.00 ©2014 IEEE

denotes on. The goal is to optimize the comfort level. There are two criteria needed to satisfy, lighting level and lighting uniformity. The former is satisfied when every sensor reading stays in an accepted range i.e. if sensor j's reading is $L_j(x)$, then $min \leq L_j(x) \leq max$. The latter can be satisfied by minimizing the standard deviation(represented by σ) of the sensor readings. Hence the problem can be stated as:

$$\begin{aligned} \underset{x}{\text{minimize}} \quad & \langle \sigma(L_1(x), ..., L_m(x)) \rangle \\ \text{subject to} \quad & min \leq L_j(x) \leq max,\ j = 1, ..., m \\ & x_i \in \{0, 1\},\ i = 1, ..., n \end{aligned} \tag{1}$$

For lighting impacts, one important feature is that sensor readings are additive [3], i.e. let $Impact_{i_j}$ to be the impact of light i on sensor j when only light i is on and it is dark outside, then we have

$$L_j(x) = \sum_{i=1}^{n} Impact_{i_j} \cdot x_i \tag{2}$$

[4] and [5] gave a general definition of nonlinear integer programming problem. It can be stated as:

$$\begin{aligned} \text{max/min} \quad & f(x) \\ \text{subject to} \quad & h_i(x) = 0,\ i \in I = 1, ..., p \\ & g_j(x) \leq 0,\ j \in J = 1, ..., q \\ & x \in \mathbb{Z}^n \end{aligned} \tag{3}$$

where x is a vector of decision variables, and some of the constraints $h_i, g_j : \mathbb{Z}^n \to \mathbb{R}$ or the objective function $f : \mathbb{Z}^n \to \mathbb{R}$ are non-linear functions. In equation 1, let $g_1(x) = L_j(x) - max, h_1(x, y) = L_j(x) - min - y = 0, y \geq 0, y \in \mathbb{Z}$. It can be transformed to:

$$\begin{aligned} \underset{x}{\text{minimize}} \quad & \langle \sigma(L_1(x), ..., L_m(x)) \rangle \\ \text{subject to} \quad & g_1(x) \leq 0 \\ & h_1(x, y) = 0 \\ & x_i \in \{0, 1\},\ i = 1, ..., n \\ & y \geq 0, y \in \mathbb{Z} \end{aligned} \tag{4}$$

Since equation (4) satisfies the format of equation (3) where the objective function is nonlinear, our problem belongs to nonlinear integer programming problem. According to [4], it is NP-hard. Therefore, any polynomial time computed solution would be an approximation. To find the best setting, a naive approach is to try all 2^n positions of n switches, which is unacceptable due to the time complexity. Therefore, we propose a heuristic algorithm for computing an approximate optimal light setting. The heuristic plan has three main steps. First, calibration, i.e. calculating $Impact_{i_j}$(section III-A). Second, sensors and lights are partitioned into small zones based on calibration data(section III-B). Third, an approximate optimal light setting is generated for each zone (section III-C).

TABLE I
CALIBRATION:LIGHTS' IMPACTS ON SENSORS(LUX)

	l_1	l_2	l_3
s_1	20	230	350
s_2	680	10	0

B. Related Work

Recently WSN technologies have been applied into a lot of areas such as [6] [7]. It consists of distributed portable wireless sensors to monitor physical conditions, such as light, temperature, humidity, etc. By using WSN, people can easily detect the change of surrounding environments and make corresponding adjustments for actuators. In lighting control area, several existing work [3][8][9] applied WSN technologies into lighting control system for energy conservation purposes. It is involved with using some specific protocols in WSN area such as ZigBee [10]. [11] proposed an approach to integrate small wireless sensor or actuator nodes into an IP-based network so that it is possible to provide web services at each node.

Several customized lighting control systems are designed for some specific areas. For example, [8] was mainly designed for entertainment and media production area while [12] was mainly designed for theater arts area. These systems generally have some specific requirements that normal lighting control systems do not have. For example, in theater arts area, it needs to capture the positions of actors in real time. Therefore, the objective functions for the lighting control in specific places need to be adjusted accordingly and some particular sensors may need to be enrolled. When more than one lighting resource are involved, such as whole lighting and local lighting, [9] proposed an approach that every user carries a sensor to detect the local intensity. [13] introduced an user-friendly interface for a networked lighting control system.

[14] presented a mathematical model for lighting control system that could be applied into the case that a luminary impact is continuous such as light emitting diodes(LEDs) luminaries and the goal illumination level is given as a single value rather than a particular range. [15] used infrared ray communication technology for smart illuminance sensor to retrieve the lighting ID binding with each lighting fixture. Through analyzing lighting ID information, sensors can recognize nearby luminaries so that the control efficiency is improved.

III. CONTROL APPROACHES

A. Calibration

Our system first determines the lighting levels, lighting locations and sensor correlations via calibration. We note that each light has an impact on a set of sensors. Calibration is used for calculating $Impact_{i_j}$. Suppose there are n lights in an area. The process of calibration is:

1) Turn off all lights and turn on one light at a time
2) Record the light's impact on each sensor
3) Repeat step 1 n times until all lights are counted

The impact data will be inserted into a table as shown in Table I. Assume there are 3 lights(l_1, l_2 and l_3) and 2 sensors(s_1 and s_2) in an area. Table I shows all lights' impacts on sensors in lux values. For example, when only l_1 is on, its impact on s_1 is 20 lux($Impact_{1_1}$) and on s_2 is 680 lux($Impact_{1_2}$).

B. Zoning/Partitioning

From the calibration data we can partition the whole area into smaller zones. The idea of zones is that some switches have no impact on some sensors as they are in different rooms or floors. We assume that every light and sensor can only belong to one zone and artificial lighting is the only lighting resource in the area.

Assume there are A sensors and B lights. For each $j, j = 1, ..., A$, we define a function $f_j : \{0,1\}^B \to \mathbb{R}_+$, which is a mapping from all lights status(on/off) to a non-negative value registered by sensor s_j. It is clear that $f_j(x) = L_j(x)$. Using the same example as shown in Table I, if l_1 is on, l_2 is off, and l_3 is on, $f_1(1,0,1) = L_1(1,0,1) = 1 \cdot 20 + 0 \cdot 230 + 1 \cdot 350 = 370$. All f_js can be collected into $F : \{0,1\}^B \to \mathbb{R}_+^A$ with for every choice of which lights are on/off gives the value of lux for all sensors.

The goal is to partition the whole area for dropping down the computational size (number of lights in a zone). For each j, $j = 1, ..., A$, we define a vector g_j of length B to indicate which lights are assigned to sensor s_j. We also set up a threshold γ, which is used to decide whether a light is partitioned into a zone or not. When only l_i is on, if the value of f_j is smaller than γ, then the ith value of the vector g_j is 0; otherwise it is 1. Proceeding with the example of Table I, we show the procedure of getting g_1 and g_2. Let's set $\gamma = 30$. Then since when only l_1 is on, the value of f_1 is 20 which is smaller than γ, 1st value of the vector g_1 is 0. The value of f_2 is 680 which is larger than γ, 1st value of the vector g_2 is 1. Applying this method, in consequence, we have $g_1 = (0,1,1)$ and $g_2 = (1,0,0)$, i.e. lights l_2 and l_3 are assigned to sensor s_1 while light l_1 is assigned to s_2.

C. Final Computation

The last step is to compute desired light settings. Note that the computations in all zones are independent and can run concurrently. We describe the final computation for a zone, with lights and sensors belonging to that zone only.

The proposed lighting control algorithm contains 4 parts. Part 1 counts the minimum number of lights needed to be turned on. Let m to be the number of sensors, $C = \{C_1, ..., C_n\}$ to be the set of lights' contributions to the whole zone where $C_i = \sum_{j=1}^{m} Impact_{i_j}$, $lowerbound$ is equivalent to $min \times m$ and $upperbound$ is equivalent to $max \times m$. To count the minimum number of lights turned on, simply find minimum number of elements in the sorted array C such that sum of them are greater or equal than $lowerbound$. It is obvious that a linear search would be enough to solve the problem. If sum of all values in C is still smaller than $lowerbound$, simply turn on all lights.

Part 2 and Part 3 compute candidates of final light setting. A candidate c is a set that only contains light numbers selected to be on, i.e. if $i \in c$ then $x_i = 1$, otherwise $x_i = 0$.

Part 2 computes the *base-level candidates*. From part 1, it is known at least w lights needed to be turned on to adjust light intensity greater or equal than $lowerbound$. Therefore, for each base-level candidate $setting$, it should contain w elements and sum of the w elements' contributions to the whole zone($\sum_{k \in setting} C_k$) should be slightly greater or equal than $lowerbound$. The proposed approach is based on the fact that lux level light intensity is additive[3] and greedy algorithm, which is shown in Algorithm 1.

Algorithm 1 Base-Level Candidates Selection

Input: $C, n, w, lowerbound$
Output: base-level candidate $setting$
1: $searchvalue \leftarrow lowerbound/w$, $s_1, s_2, s \leftarrow \emptyset$
2: find index of the value that is the smallest value larger than $searchvalue$ in C and put it in s_1
3: find index of the value that is the largest value smaller than $searchvalue$ in C and put it in s_2
4: add s_1 and s_2 to s
5: **for** $s_i \in s$ **do**
6: **for** $j = 1 \to w - 1$ **do**
7: $searchvalue \leftarrow (lowerbound - \sum_{k \in s_i} C_k)/(w - j)$
8: change s_1 to s_i in line2 and repeat
9: **end for**
10: add s_i to $setting$
11: **end for**
12: **for** $s_i \in s$ **do**
13: **for** $j = 1 \to w - 1$ **do**
14: $searchvalue \leftarrow (lowerbound - \sum_{k \in s_i} C_k)/(w - j)$
15: **if** j is odd **then**
16: change s_1 to s_i in line2 and repeat
17: **else**
18: change s_2 to s_i in line3 and repeat
19: **end if**
20: **end for**
21: add s_i to $setting$
22: **end for**
23: **return** $setting$

Part 3 generates more candidates from base-level candidates. Unlike light impacts, standard deviation is not additive, thus increasing number of lights may negatively impact the standard deviation. However, due to time and quality tradeoffs, computing more sets likely results in getting closer to the theoretical optimality. Adding δ lights to compute needs $O(n^\delta)$ time. To ensure low response time a low complexity is desirable, and based on the theoretical analysis and simulation experiments, we set $\delta = 2$. It indicates that we will at most add 2 lights to the existing candidates.

Suppose each base-level candidate has w elements, there are n lights in total. Then if each candidate wants to add an unselected light into itself, there would be $n - w$ choices.

If there are k base-level candidates, finally there would be $k \times (n-w)$ new candidates that have $w+1$ elements added. Similarly when another unselected light is trying to add into the candidates that have $w+1$ elements, total amount of candidates that have $w+2$ elements is becoming $k \times (n-w) \times (n-w-1)$. Thus total number of candidates would be $k + k \times (n-w) + k \times (n-w) \times (n-w-1)$. Note for each candidate c, $\sum_{p \in c} C_p \leq upperbound$.

Part 4 calculates the standard deviation of sensor readings generated by each candidate. According to equation 2, for each candidate c, we are able to calculate its impact on sensor j $L_j(c)$. Then it is easy to know the standard deviation of all sensors' readings under c. The candidate that generates the lowest standard deviation of sensor readings would be selected as the final light setting.

D. Time Complexity of Heuristic Lighting Control Algorithm

Suppose there are n lights, m sensors in a zone. Calibration has a time complexity $O(n)$. Zoning/Partitioning has a time complexity $O(mn)$. Final computation has 4 subparts. Part 1's time complexity is $O(n)$. The worst case for Part 2 is $O(n^2)$. As discussed earlier, the time complexity for Part 3 is n^δ. In our case, $\delta = 2$, which makes Part 3 time complexity to be $O(n^2)$. Obviously Part 4's time complexity is equivalent to Part 3, which is also $O(n^2)$. Therefore, for every zone, the total time complexity is $O(n^2)$.

When considering the whole area, suppose there are z zones, and zone $z_i (i \in [1, z])$ has n_i lights. If total number of lights is N, we have $N = \sum_{i=1}^{z} n_i$. Since computations in zones are running concurrently, the total complexity is equivalent to $O(\langle \max(n_1, ..., n_z) \rangle^2)$. It is clear $max(n_1, ..., n_z) \leq N$, so the whole area's time complexity is $O(N^2)$.

IV. Experimental Work and Simulation Results

A. Implementation Details

To demonstrate the feasibility and effectiveness of our approach, we used an experimental setup. A 8ft $\times$ 8ft test cell was instrumented with 9 lights ($15W$ incandescent, $120V$), and 9 sensors connected via a WSN and actuators to a computer. The computer is a Intel Core2 running Linux. The sensors use Crossbow's TelosB motes containing visible light sensors called Hamamatsu S1087. The sensor network uses the Collection Tree Protocol [16] to send lighting data to a designated gateway sensor. Applications running on the motes are implemented in NesC and TinyOS environment. The computer runs a server called SenServer developed in Java and connects to the gateway sensor via serial port. A ProXR Relay Controller is connected to Control Server through USB port to control artificial lights. Since the official driver of the relay controller is designed for Windows platform, Control Server is developed in C# that runs in a Windows machine. It is implemented based on the contents discussed in section III. Eclipse Standard 4.3 and Microsoft Visual Studio 2010 are used for Java and C# developments respectively.

B. Feasibility

The experiment is run at night. Each sensor is placed under each light, at a distance of 60 inches, and they are numbered 0 to 8. Experimental design and elements are shown in Fig. 1. The steps are:

1) Calibrate lights' impacts on sensors using the approach described in section III-A.
2) Set expected lux level range from 49 to 51(Test1).
3) Turn off all lights. Run control server to compute light settings. When selected lights are turned on, record the reading on each mote.
4) Change the step two's expected lux level range to be 99 to 101(Test2), 149 to 151(Test3) and 199 to 201(Test4) respectively. Rerun step three.

The results are shown in Fig. 2. From the figure, we can see for each test, the computed light setting's impact on every mote slightly fluctuates at the middle point of the expected range. Take test 3 as an example. Lights 0,1,2,3,6,7,8 are selected to be on. Every mote's reading is falling between 149 and 151. Average of mote readings for test 3 is 149.22, standard deviation is 3.80.

All other tests show the similar trends, which shows the feasibility of our real system, that is using the results of our computation and then applying it to a real, calibrated room, it provides the results that we predicted. Prior to this test, we had already measured the additive property and had some experience with daylight measurements.

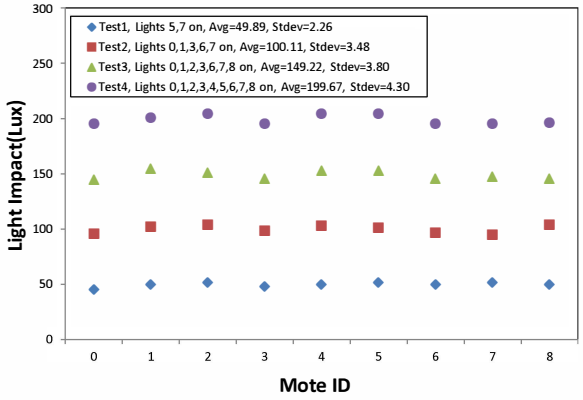


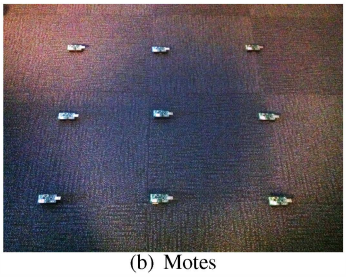
Fig. 2. Experimental Results of 9 lights and 9 motes

C. Adaptivity and Scalability

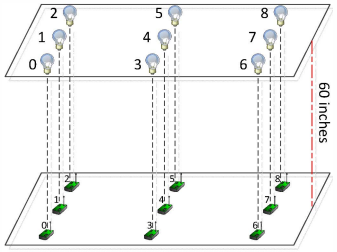
The test cell experiments are limited, as the number of lights are low, the space is limited and the geometry is rather simple. To be able to study adaptivity and scalability we did simulations on a more complex, synthetic setup, with randomly generated impact values.

Adaptivity and scalability are two important features of the system. Adaptivity means when room pattern changes such as adding or removing some lights, changing lights' positions, etc, the system can automatically detect environmental changes and make corresponding adjustments. In other words, when room pattern changes, the lighting control algorithm

(a) Lights



(b) Motes



(c) Testbed Layout

Fig. 1. Experimental Design

should be able to adjust light settings accordingly in a short time without any modifications. Adaptivity is also necessary when there is additional illumination due to outside lights. Scalability means when amount of lights is growing large, the lighting control algorithm could still compute the light settings in a reasonable time. Apparently customized algorithms do not have the adaptivity feature and custom systems also use brute force methods to determine lighting levels and hence have exponential complexity, thus not being scalable.

When pattern changes, the only part that might get changed in the objective function(Equation 1) is $Impact_{i_j}$. Therefore, to test if the heuristic algorithm is adaptive, $Impact_{i_j}$ should be randomly assigned. In addition, expected lux level for this experiment is set between 320 and 500 lux to match normal office lighting range [17]. The simulation experiment runs as follows:

1) Assume there are 30 lights and 25 sensors in a zone. Each light i has an impact value on sensor $j(Impact_{i_j})$. There should be 750(30×25) impact values.
2) Randomly assign an integer value from 0 to 100(specification of $15W$, $120V$ light bulb) to be one impact value. Repeat this step 750 times until all impact values are successfully assigned.
3) Run brute force algorithm($O(2^n)$), the proposed lighting control algorithm($O(n^2)$), and a revised $O(n^3)$ algorithm($\delta = 3$) that adds one more light to the candidate sets of $O(n^2)$, respectively. Record computational time, lux level light intensity and standard deviation for each method.
4) Repeat step 1-3 1000 times. Take an average of standard deviation, light intensity and computational time results respectively.
5) Increase light's amount and rerun steps 1-4.

The result is shown in Fig. 3. Fig. 3(c) describes time performances of different methods. Since when amount of lights increases, the computational time of brute force method is increasing exponentially, which quickly arrives at a very large point of value while polynomial time algorithms have a pretty low computational time. Therefore, in order to put all data together, for Fig. 3(c), we take a common logarithm(log_{10}) on the real computational time data. From the graph, we can see that our proposed $O(n^2)$ algorithm reaches a similar performance in light intensity and standard deviation compared to the revised $O(n^3)$ algorithm, but the computational time is far less than the latter one. Compared to the brute force, the proposed approach has an extremely better time performance with a similar light intensity performance and a reasonable increase on standard deviation.

We also do some other evaluations to verify the system's scalability. For example, we run the above algorithms on very large dataset like 1000 number of lights. Result shows that the proposed $O(n^2)$ approach can still compute the light setting in a desired range while other methods are not applicable due to the time performance. When size is 1000, the $O(n^2)$ algorithm is running in around 10^6 time complexity which reduces substantial workloads from brute force's 1.07×10^{301} time complexity.

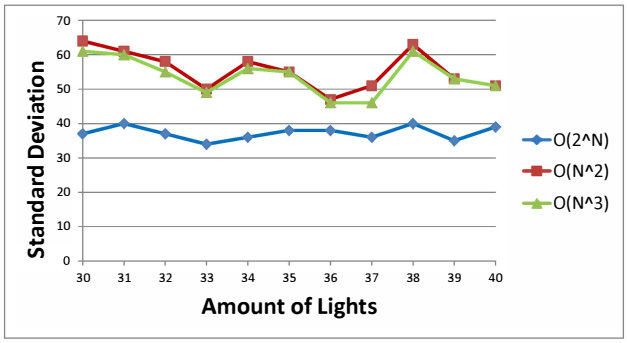
V. Extending the Algorithm

Our goal is to produce a stable and uniform lighting environment in workplaces by adjusting artificial lights. In this paper, we assume artificial lighting is the only lighting source in a workplace and there is no other lighting involved. However, when ambient light exists, the algorithm needs to be extended to allow for the impact produced by such light. Ambient light will produce a non-zero and varying impact on different sensors and is not controllable via light switches. This impact will have to be detected and used in the computation, in other words the computation needs to compute deltas between current conditions and desired levels and recomputed switch settings. The idea can be also applied when required illumination levels in an area are different.

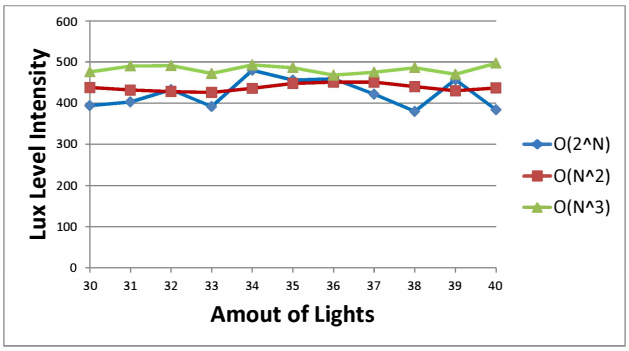
The selection of the final computation results can be done in an energy efficient manner by computing the energy costs of each light being turned on. From this we can select candidate setting that may not produce the best uniformity but its energy is consumed in significantly lower. In addition, the goal lighting level can be changed for energy savings especially at peak load hours.

Placement of sensors is an important point we did not address in the paper. In the current work, we assume sensors in an area are distributed in a good way that can very well capture the light intensity. In the future work, we will do more investigations on sensor layouts, which will better capture the light intensity in an area.

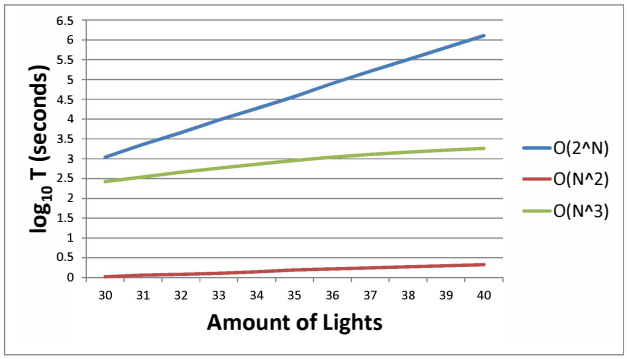
Although the brute force method can't be directly applied due to the time issue, it still might be worthy to add into the



(a) Standard Deviation



(b) Light Intensity



(c) Computational Time

Fig. 3. Comparison of Different Approaches

system because it can produce the theoretical best result. One idea is to combine our current work with brute force method. The current work can be used during the initial step to quickly generate an accepted light setting. The brute force method can finely adjust the setting in the backend afterwards. Optimal results can be also stored for future use.

VI. Conclusion

To enable automated lighting control, under varying conditions of occupancy, needs, outside lighting influences and other perturbations it is essential to have a core algorithm that is effective and adaptive. Such algorithm must be deployable in a simple, cost effective system without the need for customizations and reprogramming as conditions change. This paper presents such a core algorithm and tests its effectiveness in both experimental and simulated scenarios. The underlying system of wireless sensors and wired actuators can be deployed at homes and in larger building without excessive costs. We build a formal model of the lighting control problem and show it is a hard problem (NP-Hard). A heuristic algorithm is proposed to solve the problem to compute approximate optimal solution. Experimental results show the efficiency and effectiveness of the heuristic algorithm. The proposed approach can be augmented to accommodate daylighting, failures, manual overrides, and occupancy detectors.

References

[1] *Lighting at work*. Health and Safety Executive, 1997.

[2] R. Mohamaddoust, A. T. Haghighat, M. J. Motahari Sharif, and N. Capanni, "A novel design of an automatic lighting control system for a wireless sensor network with increased sensor lifetime and reduced sensor numbers," *Sensors*, vol. 11, no. 9, pp. 8933–8952, 2011.

[3] V. Singhvi, A. Krause, C. Guestrin, J. Garrett Jr, and H. Matthews, "Intelligent light control using sensor networks," in *Proceedings of the 3rd international conference on Embedded networked sensor systems*. ACM, 2005, pp. 218–229.

[4] R. Hemmecke, M. Köppe, J. Lee, and R. Weismantel, "Nonlinear integer programming," *arXiv preprint arXiv:0906.5171*, 2009.

[5] Wikipedia. Nonlinear programming.

[6] A. Mainwaring, D. Culler, J. Polastre, R. Szewczyk, and J. Anderson, "Wireless sensor networks for habitat monitoring," in *Proceedings of the 1st ACM international workshop on Wireless sensor networks and applications*. ACM, 2002, pp. 88–97.

[7] Q. Li, M. De Rosa, and D. Rus, "Distributed algorithms for guiding navigation across a sensor network," in *Proceedings of the 9th annual international conference on Mobile computing and networking*. ACM, 2003, pp. 313–325.

[8] H. Park, J. Burke, and M. B. Srivastava, "Design and implementation of a wireless sensor network for intelligent light control," in *Proceedings of the 6th international conference on Information processing in sensor networks*. ACM, 2007, pp. 370–379.

[9] M. Pan, L. Yeh, Y. Chen, Y. Lin, and Y. Tseng, "Design and implementation of a wsn-based intelligent light control system," in *Distributed Computing Systems Workshops, 2008. ICDCS'08. 28th International Conference on*. IEEE, 2008, pp. 321–326.

[10] Y. Wang and Z. Wang, "Design of intelligent residential lighting control system based on zigbee wireless sensor network and fuzzy controller," in *Machine Vision and Human-Machine Interface (MVHI), 2010 International Conference on*. IEEE, 2010, pp. 561–564.

[11] L. Schor, P. Sommer, and R. Wattenhofer, "Towards a zero-configuration wireless sensor network architecture for smart buildings," in *Proceedings of the First ACM Workshop on Embedded Sensing Systems for Energy-Efficiency in Buildings*. ACM, 2009, pp. 31–36.

[12] C. Feng, L. Yang, J. W. Rozenblit, and P. Beudert, "Design of a wireless sensor network based automatic light controller in theater arts," in *Engineering of Computer-Based Systems, 2007. ECBS'07. 14th Annual IEEE International Conference and Workshops on the*. IEEE, 2007, pp. 161–170.

[13] T. Hiroyasu, A. Nakamura, S. Shinohara, M. Yoshimi, M. Miki, and H. Yokouchi, "Intelligent lighting control user interface through design of illuminance distribution," in *Intelligent Systems Design and Applications, 2009. ISDA'09. Ninth International Conference on*. IEEE, 2009, pp. 714–719.

[14] A. Schaeper, C. Palazuelos, D. Denteneer, and O. Garcia-Morchon, "Intelligent lighting control using sensor networks," in *Networking, Sensing and Control (ICNSC), 2013 10th IEEE International Conference on*. IEEE, 2013, pp. 170–175.

[15] M. Miki, A. Amamiya, and T. Hiroyasu, "Distributed optimal control of lighting based on stochastic hill climbing method with variable neighborhood," in *Systems, Man and Cybernetics, 2007. ISIC. IEEE International Conference on*. IEEE, 2007, pp. 1676–1680.

[16] O. Gnawali, R. Fonseca, K. Jamieson, D. Moss, and P. Levis, "Collection tree protocol," in *Proceedings of the 7th ACM Conference on Embedded Networked Sensor Systems*. ACM, 2009, pp. 1–14.

[17] Wikipedia. Lux. [Online]. Available: http://en.wikipedia.org/wiki/Lux