A Negotiation Approach for Energy-aware Room Allocation Systems

Sergio Esparcia[1], Victor Sánchez-Anguix[1], Reyhan Aydoğan[2]

[1] Departamento de Sistemas Informáticos y Computación
Universitat Politècnica de València
Camino de Vera s/n, 46022 - Valencia (Spain)
{sesparcia,sanguix }@dsic.upv.es
[2] Interactive Intelligence Group
Delft University of Technology
Delft, The Netherlands
R.Aydogan@tudelft.nl

Abstract. This paper addresses energy-aware room allocation management where the system aims to satisfy individuals' needs as much as possible while concerning total energy consumption in a building. In the problem, there are a several rooms having varied settings resulting in different energy consumption. The main objective of the system is not only finding the right allocations for user's need, but also minimizing energy consumption. However, the users of the system may have conflicting preferences over the rooms to be allocated for them. This paper pursues how the system can increase user satisfaction while achieving its goals. For that purpose, an adaptation of the mediated single text negotiation model is introduced. The proposal seeks to guarantee an upper bound on energy consumption by pruning the negotiation space via a genetic algorithm, and to take advantage of the negotiation for increasing user satisfaction. Experiments suggest that the adaptations improve the performance.
Key words:automated negotiation,room allocation,energy consumption

1 Introduction

The primary energy consumption in the world in 2004 has been reported as 11059 MToe (Million of Tonnes of Oil Equivalent), while it was 6034 MToe in 1973 [6]. That shows that we are consuming more and more energy over time. Due to the decreasing of the amount of the energy resources and the environmental effects of excessive energy consumption, reducing energy consumption has become a critical issue.

Buildings are one of the main sources of energy consumption, and its consumption is increasing rapidly. For example, in 2004, in the European Union, buildings consumed 37% of the final energy, more than transport, whose consumption was a 32%, and the industry, which consumed a 28% of the total energy consumption. In this territory, the energy consumption increases 1.5% each year [6]. Therefore, it seems clear that keeping the energy consumption of a building under control is important for the environment of our planet. This leads to the need for the emergence of energy-aware systems aiming to minimize the energy consumption inside buildings. For example, the Government of the Region of Valencia (Spain) established the minimum air conditioning temperature to 26 centigrade degrees at workplaces during summer in order to reduce the electricity consumption [1]. Although this rule is aimed to reduce the energy consumption and consequently to save the environment and reduce energy expenses, it may not satisfy everyone at the same degree in the society. For example, this temperature may be too hot for some workers who have difficult working conditions under very warm summer days. Therefore, it is desired to meet the individuals' preferences at the same time while reducing the energy consumption.

In this paper, we address the problem of energy-aware scheduling and resource allocation. Our primary aim is to minimize the energy consumption, but we also desire to satisfy the individuals' preferences as much as possible. The proposed system is based on the assignment of activities around different rooms of a building. This case-study is a revision of the scenario presented in [10]. The building belongs to a university, and its rooms are dedicated to a certain purpose (teaching, meeting, etc.). Thus, they have different equipments that allow them to carry out different purposes. The activities are managed by the building manager, who is responsible for distributing the different activities. The activities are sent to the building manager by staff members that require a room with specific features with a start and end time. The goal of the building is finding a proper room allocation for the requested activities. However, the number of available spots for an activity are limited. Therefore, it is necessary to solve this problem by taking into account the constraints imposed by the petitions and the characteristics of the different rooms. Due to differences between rooms, users have different preferences. For example, a user may be interested on a specific room because of different reasons, like the location of the room (the closer to his office, the better), its capacity (not too big, not too small), the acoustics and luminosity inside this room, the quality of the equipment (a user may look for new computers or other new equipments), etc.

A classic solution to this problem is applying some optimization algorithm, such as a genetic algorithm. However, the preferences of the participants may not be known and they may be patients. In such a case, the applied optimization technique will produce a solution by only considering the system's preferences, minimizing energy consumption. Therefore, it is necessary to take into account the agents' preferences when allocating the activities. In order to deal with this problem, we present a multiparty negotiation approach in which the system generates possible room allocations, and asks to each participant whether or not it is acceptable for them, and refines its offer based on their feedback. It is worth noting that the preferences of the participants are kept as private. The proposed approach is a variant of the mediated single text negotiation protocol [3] adapted to the present domain. Using negotiation presents the following advantages: (i) each user participates as a software agent that keeps its preferences private; (ii) negotiation is a "human" process (i.e. it is understandable by people) where humans can directly participate at some point; and (iii) allocations can be renegotiated, which allows systems to adapt themselves to changes.

The rest of this paper is structured as follows: Section 2 describes the case-study and the problem that is intended to be solved. Section 3 explains the negotiation model that is followed to solve the previously presented scenario. Section 4 presents the results of the experiments carried out using this negotiation model. Finally, Section 5 depicts the related work on this topic, as well as the conclusions and future work on this topic.

2 Problem description

This case study focuses on how to manage the petitions of activities to be carried out in the different rooms of a building in a university. The type of activities that can be carried out inside a room are limited by two factors: (i) the capacity of the room, and (ii) the equipment of the room. The building is open during weekdays (from Monday to Friday) and closes on weekends. The working hours of the building are from 8:00 to 20:00. The activities take place in slots of 60 minutes, starting at the minute :00 each hour and finishing at the :59 of the same hour. The activities can last from one to n slots.

The scheduling of activities in the rooms is controlled by the building manager. The users of the building (the agents that are responsible for each activity) send their petitions to the building manager, who is responsible for managing petitions and finally making public an adequate schedule. A petition includes the number of people that will participate in the activity, the type and quantity of equipment they require, the start time of the activity, and the number of slots of an hour that they need.

Additionally, there is another requirement imposed by the building manager that has to be taken into account. Its objective is keeping energy consumption at a reasonable limit (i.e. minimize energy consumption). The energy consumption is calculated as the sum of the consumption of the whole set of rooms. Then, the consumption of each room is the sum of the individual consumption for each unit of equipment used. The system is a multiagent system described as $MAS = \langle R, A, E, s \rangle$, where R is the set of rooms of the building, A is the set of agents representing the users that make petitions, E is the set of the different equipment that a room may have, and s is the building manager.

2.1 Rooms

A room $r_1 \in R$ is defined by means of three features: its capacity (ca), the number of units per equipment type it has (h), and the average consumption per hour of each type of equipment (co). These features are represented as follows:

$$\begin{aligned} ca &: R \rightarrow \mathbb{N} \\ h &: R \times E \rightarrow \mathbb{N} \\ co &: R \times E \rightarrow \mathbb{R} \end{aligned} \tag{1}$$

where the function ca defines, given a room $r_i \in R$, the number of people that fit inside that room. Given a room $r_i \in R$ and a type of equipment $eq_j \in E$, the function h returns the number of units that the room is equipped with. In the case the room is not equipped with a specific type of equipment, the function returns 0. Finally, the function co returns the consumption (in kW/hour) that a unit of equipment of the type $eq_j \in E$ makes during an hour in the room $r_i \in R$.

2.2 Agent

An agent $a_i \in A$ is represented by means of the petition he makes. This petition includes the number of participants in the activity the agent is requesting (pa), the number of units of each type of equipment that are required (req), the date and time when the activity has to start ($date$), and the length of the activity (len):

$$\begin{array}{l} pa : A \rightarrow \mathbb{N} \\ req : A \times E \rightarrow \mathbb{N} \\ date : A \rightarrow T \\ len : A \rightarrow \mathbb{N} \end{array} \tag{2}$$

where pa returns, given an agent $a_i \in A$, the number of participants on the activity requested by the agent. The function req will return the number of units of this type of equipment that the agent requests, given an agent $a_i \in A$ and a type of equipment $eq_j \in E$. In the case of $date$, the moment in time $t \in T$ is represented as:

$$t = \langle day, time \rangle \tag{3}$$

where $day \in \{mon, tue, wed, thu, fri\}$ is the day of the week to carry the activity out, and $time \in [8, 19]$ represents the hour when the activity in the petition is intended to start. Finally, len returns the number of hours required by the activity.

2.3 Room allocation

A room allocation is an assignment of one room per petition $X = (x_1, x_2, x_3, ..., x_A)$ being $x_i \in R$ is the room assigned to agent a_i. It represents a possible solution for the problem presented in this paper. During this work, we will talk indistinctly of room allocation, solution, or offer, since we consider them to be the same object. In order to be valid, the solution has to satisfy the following constraints:

- $\forall x_i \in X, eq_j \in E : h(x_i, eq_j) \geq req(a_i, eq_j)$
  – The room has the required units of equipment requested in the assigned petition.
- $\forall x_i \in X, a_i \in A : ca(x_i) \geq pa(a_i)$
  – The number of participants is lower than the capability of the assigned room.
- $\forall x_i, x_j \in X : \neg overlap(x_i, x_j)$
  – There is no overlap between activities. The function $overlap(x_i, x_j)$ returns $true$ if there is an overlap between the activities requested by agents a_i and a_j.

3 Negotiation Model

Our proposal takes inspiration from the mediated single text negotiation protocol presented in [3]. In Klein *et al.*, the mediator initially generates a random bid and asks the negotiating agents to vote for this bid. Each agent can vote to either “accept” or “reject” according to its negotiation strategy. If all negotiating agents vote to accept, the bid is selected as the most recent mutually accepted bid. In further rounds, the mediator modifies the most recent mutually accepted bid by randomly exchanging one value with another one in the bid and queries negotiating agents to vote for the current bid. The negotiation process continues iteratively until a predefined number of rounds is reached. The next section describes the process followed to reach an agreed room allocation, depicting how we adapted the model proposed by Klein *et al.* to our domain.

3.1 Modeling preferences

Agents' preferences Agents have different preferences over room assignments (e.g., walking distance, acoustics, equipment quality, etc.). Thus, as users of the reservation system, agents not only want to have their petitions fulfilled, but also being assigned their most preferred rooms if possible. Given a room allocation X, an agent's satisfaction is only affected by the room which has been assigned to its activity. Thus, the utility function for agent a_i can be defined as $U_{a_i}(X) = sat_i(x_i)$, where $sat_i(x_i)$ defines the satisfaction generated by being assigned a certain room. The function is scaled to $[0,1]$ so that 1 represents the highest satisfaction and 0 designates the lowest satisfaction obtainable with a room assignment. Each agent aims to maximize its own utility.

Building manager preferences The building manager has the goal of managing agents' petitions and providing an adequate room allocation for the activities that have been requested. However, given the increasing importance of energy consumption in our society, not every room allocation is considered as adequate. Instead, the building manager scores the different room allocations based on the energy consumption stemming from the room assignment. As a manager, his prime goal is minimizing energy consumption. The energy consumption associated to the room allocation X can be defined as:

$$EC_s(X) = \sum_{i=1}^{|A|} \sum_{eq_j \in E} req(a_i, eq_j) * co(x_i, eq_j) * len(a_i) \tag{4}$$

Basically, it is defined as the sum of each petitions' consumption, where a petition's consumption can be defined as the sum of the consumption generated by the required equipment units during the assigned time.

3.2 Proposed negotiation approach

As stated, the negotiation protocol is a variant of the mediated single text negotiation protocol proposed by Klein *et al.* [3] adapted to our domain. The original protocol is based on the existence of a mediator that helps agents reach an agreement by mutating the most recent mutually accepted offer. In our case, the building manager samples a set of low energy consumption allocations by means of a genetic algorithm (GA) before the negotiation. The reasons to this genetic sampling are: (i) the building manager needs to look for good solutions that satisfy the main goal of the system, energy consumption; (ii) the domain is extremely huge (e.g., even for a *small* problem like $|R| = 10$ and $|A| = 10$ the search space is 10^{10}); (iii) employing a metaheuristic like a GA provides an anytime optimization. During the negotiation, the building manager plays the role of mediator and it proposes offers with low energy consumption from the sample obtained from the GA. Hence, a bound on the energy consumption (i.e., a threshold) can be assured, fulfilling the main goal of the system. Then, the negotiation aims to satisfy agents' preferences as much as possible with proposals that are below the aforementioned threshold.

The negotiation protocol goes as follows. In the first negotiation round, the building manager informs of the initial agreement for the group, which is automatically accepted. From that point on, the manager proposes one offer per round to the group. This offer is then sent to the agents, who cast their votes (either 'better', 'worse', or 'same'). Depending on the specific mechanism employed by the manager, which will be explained later, the offer is accepted or rejected. If the offer is accepted, the manager informs the rest of agents and the offer becomes the most recently accepted solution. The negotiation continues following this iterative process until a number of negotiation rounds N is reached. In the end, the final agreement is the most recently accepted solution.

Agent's Voting Mechanism During the negotiation agents receive offers from the building manager and they are asked to vote. By default, the first offer proposed by the building manager becomes the initial agreement. In following negotiation rounds, agents can give feedback. Considering that X is the last offer accepted by the group, an agent a_i emits a 'better' vote if $U_{a_i}(X) < U_{a_i}(X')$, it emits a 'worse' vote (i.e., reject) if $U_{a_i}(X) > U_{a_i}(X')$, and it emits a 'same' vote if $U_{a_i}(X) = U_{a_i}(X')$.

Pre-Negotiation: Sampling and defining the negotiation space When the building manager engages agents to agree on a room allocation, it does not suffice with just calculating the optimal solution from the point of view of energy consumption. In fact, it may be likely that the best solution in terms of energy consumption does not generate a high user satisfaction for petitioners. Instead, a set of high quality and significantly different room allocations is needed to look for possible win-win situations. Additionally, constraints imposed over the search space by rooms' limitations and petitions' requirements preclude the building manager from employing simple optimization methods like the ones used to generate new solutions in monotonic search spaces. For that matter a niching genetic algorithm [8, 5] is employed to sample solutions before the negotiation starts. Niching genetic algorithms are a special type of genetic algorithm that avoids converging towards a single high quality solution. Instead, the population converges towards different, yet high quality solutions. This effect is accomplished by introducing local competition among similar solutions. Niching genetic algorithms have been used to tackle other complex search spaces in negotiation [7].

Before the negotiation starts, the building manager calculates a set of high quality room allocations in terms of energy consumption. The niching genetic algorithm has the following characteristics:

– Chromosome representation and fitness function: Solutions are represented as a vector of integers, where each vector index represents a petition and its content represents the room where the petition is assigned. For the fitness function, the energy consumption function introduced in Equation 4 is employed. The goal of the genetic algorithm is minimizing the fitness function.

– Initial population: Initially, a population of room allocations is randomly generated. The population has a maximum size of MAX_POP.
– Genetic operators: Crossover and mutation operators are applied over pairs of the genetic pool. More specifically, $\frac{MAX\_POP}{2}$ disjoint pairs of solutions are randomly formed. Genetic operators are applied over such pairs.
  • Crossover operator: It is applied over two parents X_i and X_j, and it generates two different children X_i' and X_i'. It operates on a chromosome per chromosome basis (i.e., attribute per attribute). With an equal probability and given a certain chromosome index k, the chromosome k from X_i is inherited by X_i' and the chromosome k from X_j is inherited by X_j'. Otherwise the chromosome k from X_i is inherited by X_j' and the chromosome k from X_j is inherited by X_i'. The probability of applying a crossover operator over a pair of solutions is controlled by the p_{cross} parameter.
  • Mutation operator: It is applied over one parent and it produces one child. The mutation operator goes chromosome per chromosome and with a probability p_{matt}, it changes the value of the current chromosome to other valid value.
– Selection Operator: It decides the solutions that are to be part of the next generation. It introduces the niching effect by inducing competition between similar solutions. The selection operator is used after each crossover and mutation operation. In the case of the mutation operation, it takes the parent and the child and it decides which one takes part in the next generation. In the case of a crossover operation, the selection operation takes both parents and both children. Each parent is paired up with its most similar child, and then, the selection operator is applied two decide whether the parent or the child go to the next generation. The selection operator is composed of a portfolio containing a deterministic crowding rule $DC(X_1,X_2)$ and a probabilistic crowding rule $PC(X_1,X_2)$. The deterministic crowding rule selects always the solution with the best fitness, whereas the probabilistic crowding rule allows for worse fitness solution to pass to the next generation with a small probability for escaping local minima/maxima. The use of of deterministic crowding and probabilistic crowding is controlled by p_{pc}, which refers to the probability of applying probabilistic crowding rules. The probabilistic crowding rule can be defined as:

$$PC(X_1,X_2) = \begin{cases} X_1 \text{ if random()} \geq p_1 \\ X_2 \text{ otherwise} \end{cases} \tag{5}$$

with $p_1 = \frac{EC_s(X_1)}{EC_s(X_1)+EC_s(X_2)}$.
– Stop criteria: The genetic algorithm stops its execution when the fitness of the best solution has not improved in MAX_IT iterations.

Since energy consumption is a prime goal for the system, the building manager may prune from the negotiation space all those solutions that are not the best from an energy perspective. From the sample obtained by the genetic algorithm, the building manager discards all of those solutions whose energy consumption is over a certain threshold. More specifically, if X_{best} is the best solution obtained by the GA, the building manager will discard those solutions X that $EC_s(X) > EC_s(X_{\text{best}}) \times tol_s$, where $tol_s > 1$ and it represents the tolerance of the building manager with respect to energy consumption.

Therefore, the building manager dictates that any agreement found in the negotiation can be no worse than tol_s times the best solution found by the GA. The set of solutions NS from the GA population that are below the threshold become the negotiation space.

Once the negotiation space has been defined by the manager, the manager scales his own utility function as follows:

$$U_s(X) = 1 - \frac{EC_s(X) - EC_s(X_{best})}{EC_s(X_{best}) \times tols_s - EC_s(X_{best})} \tag{6}$$

Negotiation: Proposing and accepting solutions In this paper we introduce special features in the offer proposal and the offer acceptance mechanism to adapt it to our domain. Regarding the offer proposal mechanism, the first offer proposed to the group, which is accepted by default, is the one with the highest utility (i.e., lowest energy consumption) for the building manager. In the next rounds, the offer proposal takes into account the agents' feedback during the negotiation. Since the most recently accepted solution implicitly represents the feedback given by the agents, the building manager proposes an offer from NS (i.e., the filtered sample from the GA), which is the most similar[3] to the most recently accepted solution. Given the fact that an agent is only affected by the attribute corresponding to its room allocation, the manager can keep track of those room assignments that are worse for each agent than the allocation in the most recently accepted solution. For instance, if x_i is the current allocation for the petition made by agent a_i in the most recently accepted offer and the manager proposes x_i', which receives a 'worse' vote by a_i, then any other solution with x_i' assigned for the petition from a_i will be rejected by the agent. Therefore, when selecting the most similar offer from NS, it skips those solutions that contain values that are known to be worse for each agent's current allocations. If all the solutions from NS are skipped, then the similarity mechanism takes into account every solution in NS.

As for the acceptance mechanism employed by the building manager, we contemplate two options in our initial study similarly to Klein *et al.* [3]:

- Hill Climber Manager: It updates the most recently accepted solution to X when no petitioner emits a 'worse' vote over X (it can emit 'same' votes and be accepted).
- Annealer Manager: It may accept an offer even if some negative votes are received. More specifically, it assigns a probability of acceptance $p_{ac}(X) = e^{\frac{-dE}{T}}$, where T is the virtual temperature which will be declined over time. In this study, we take the remaining number of rounds scaled to (0,1]. $dE = \frac{nw}{|A|}$, with nw being the number of 'worse' votes received for the offer X. According to this formula, the building manager is likely to accept an offer X as "most recently accepted offer" if the proportion of 'worse' votes over all votes is low in the beginning. Over time, this probability for the same proportion of 'worse' votes will decrease and the offer becomes unacceptable. In other words, the higher remaining time to complete the negotiation, and the smaller proportion of negative votes, the greater the probability that the the offer X will be accepted even though there are some negative votes for that offer. While approaching the deadline, annealer gradually declines so eventually he will

[3] Manhattan distance is used

act as a hill-climber and only accepts the offers as most recently accepted offer if all agents use positive vote for them. We restrict the annealing probability only to those offers which received more positive votes than negatives to assure that at least a higher number of petitioners benefit from the new offer. Otherwise the offer is automatically rejected.

4 Experimental Evaluation

In this section, we evaluate the performance of the proposed model in 90 different scenarios. We randomly generated test cases considering a number of rooms $|R|$ equal to 10, 20, or 30. The number of petitioners $|A|$ was set to be 10, 20, and 30. For every possible combination of $|R|$ and $|A|$, 10 different scenarios were generated. Hence, the total number of generated test cases was $3 \times 3 \times 10 = 90$ (possible number of rooms $\times$ possible number of petitions $\times$ number of different scenarios). The size of the smallest test cases ($|R| = 10$, $|A| = 10$) is 10^{10} with 10 agents, while the size of the largest test cases ($|R| = 30$, $|A| = 30$) is 30^{30} with 30 agents.

In order to assess the quality of the final agreement, we use a social welfare measure that sums up the utility of the building manager and the agents. We consider that the highest the social welfare, the better performance the model achieves.

$$SW(X) = U_s(X) + \sum_{a_i \in A} U_{a_i}(X) \tag{7}$$

In our experiments, we compared the performance of our hill climber manager (HC), the performance of our annealer manager (AN), the performance of the best solution found by the genetic algorithm (i.e., no negotiation carried out after the optimization), and the performance of a basic annealer manager (BA). The basic annealer does not keep track of which room values are worse than the current room allocation for each agent. The proposal mechanism just selects the offer which is the most similar to the most recently accepted solution. Additionally, the basic annealer may accept an offer (attending to $p_{ac}(X) = e^{\frac{-dE}{T}}$) even if the number of 'worse' votes received is greater than or equal to the number of 'better' votes. Hence, differently to our annealer manager, it does not assure that the new solution benefits at least more members than it detriments. The inclusion of the basic annealer manager aims to assess the benefits of the adaptations proposed to tackle the present domain. Next, we describe the experiments that we carried out.

First, we measured the social welfare in all of the test cases when the number of negotiation rounds is $N = 100$. The execution of each scenario was repeated four times to capture stochastic differences. The parameters[4] for the genetic algorithm were set to $MAX\_POP = 8192$, $p_{\text{cross}} = 70\%$, $p_{\text{matt}} = 10\%$, $p_{\text{pc}} = 90\%$, and $MAX\_IT = 100$. Additionally, the tolerance of the building manager was set to $tol_s = 1.10$. Therefore, the building manager assures that the final agreement is at most 10% worse than the best solution in terms of energy consumption found by the GA. The same pool generated by the GA was provided to the hill climber, the annealer, and the basic annealer

[4] An initial grid search was performed to find good parameters for the GA

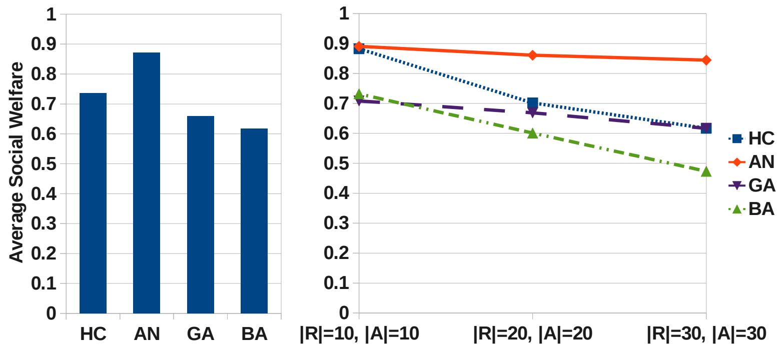

Fig. 1. Average social welfare when the number of negotiation rounds is $N = 100$ for all the test scenarios (left), and average social welfare for increasing problem sizes and agents (right).

managers. Fig. 1 shows the average social welfare scaled to [0,1] [5] in all of the scenarios (left side). In order to study the effect of the problem size and the number of agents on the negotiation, we also plotted the average social welfare for test cases with $|R| = 10$ and $|A| = 10$, $|R| = 20$ and $|A| = 20$, and $|A| = 30$ and $|R| = 30$ (right side). It can be observed in the left part of Fig. 1 that our annealer manager is the one that achieves the highest average social welfare (0.87), followed by our hill climber manager (0.73), the genetic algorithm (0.65), and the basic annealer (0.62). The performance obtained by the proposed annealer manager is close to the optimal social welfare obtainable from the genetic pool. An ANOVA test with Bonferroni post-hoc analysis suggested that the differences between the methods are statistically significant. Surprisingly, the basic annealer mediator is slightly worse than using the GA, which suggests that using our heuristic approaches in the proposed annealer manager is more adequate for the case study. When observing the right side of Fig. 1, it can be observed that, as the size of the problem increases and more agents are involved, the performance of the genetic algorithm, the basic annealer, and the hill climber manager decreases. Contrarily, the performance of the annealer manager is only slightly affected. This graphic suggests that the performance of the annealer manager is more robust to the number of negotiating agents and the problem size than the other methods, which is also highly desirable.

Finally, in the second experiment we measured the performance of the annealer manager for different maximum number of negotiation rounds. More specifically, N was set to 25, 50, 100, 200, and 400. The rest of parameters was set to the same value as the previous experiment. The results of the experiment are shown in Fig. 2. As it is can be seen, the average social welfare obtained by the annealer manager increases as the number of negotiation rounds increases. It approximately converges to a maximum value when $N = 100$. From that point, even quadrupling the number of negotiation rounds does not have any effect on the average social welfare. This convergence suggests that additional changes in the core of the annealer manager (e.g., proposal method, acceptance method, etc.) may be necessary to improve its performance. If compared

[5] 0 being the lowest, and 1 being the highest social welfare obtainable from the pool.

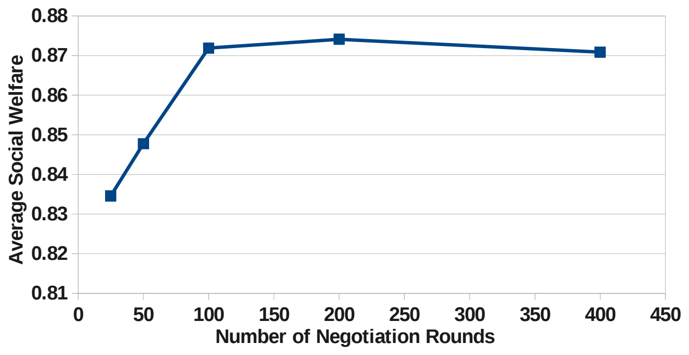

Fig. 2. Social welfare of the annealer manager at the end of the negotiation for different rounds.

with the average social welfare obtained by other methods in the first experiment, the annealer manager outperforms the rest even when the number of rounds is four times lower (i.e., 0.83 when $N = 25$ versus 0.73 obtained by hill climber manager, 0.65 obtained by the GA, and 0.62 obtained by the basic annealer when $N = 100$). Therefore, it is possible to reduce communication costs and still obtain a higher performance, which is quite desirable in case that a quicker solution is needed.

5 Discussion and Future Work

In this paper, we address the problem of energy-aware room allocation in a building. The system's goal is obtaining an adequate allocation that minimizes consumption while satisfying users' preferences as much as possible. For this matter, we introduce a variant of the mediated single text negotiation protocol [3]. The main differences reside in the fact that (i) the manager desires to minimize energy consumption and it prunes the negotiation space accordingly by means of a genetic algorithm and a threshold; (ii) agents' votes are expressed in three categories that describe their preference with respect to the new allocation ('worse', 'better', 'same'); (iii) since some solutions are not valid, the manager proposes new offers based on a similarity mechanism that takes into account those rooms that do not improve the current agreement for each agent; and (iv) the annealer mechanism ensures that offers are only accepted only if more agents benefit than agents detriment with the new offer. The experiments show that the adaptations carried out in our annealer manager are more adequate for the domain at hand.

Kuo *et al.* [4] propose the use of linear programming for planning and scheduling the time of an operation room. In order to do so, there are taken into account features such as the type of surgery (vascular, trauma, etc.), the time for each specific operation, the economic costs, etc. However, they only focus on maximizing operational issues.

Similarly to our approach, Schumann *et al.* [9] take into account the preferences of the users to maximize their termal comfort while reducing energy consumption. They adjust the thermal control by means of the predicted preferences of agents. In addition to minimizing the energy consumption, we also deal with room allocation.

One of the studies presenting a multiagent system (MAS) approach for energy consumption is that Multi-Agent Home Automation System (MAHAS) [2], a MAS that is used to manage the energy consumption of a house. MAHAS' agents are capable of calculating the consumed energy as well as predicting energy consumption. In this case, MAHAS presents two mechanisms: reactive and anticipative. The reactive mechanism protects the house from violations of energy constraints and it guarantees a good level of inhabitant satisfaction. The anticipative mechanism computes plans for production and consumption of services in a house, anticipating the energy consumption for the devices inside the house. In addition to energy consumption, in our problem we consider room allocation where constraints exist on which activities can be hosted in each room.

Some future lines of work include considering the robustness of solutions with respect to unexpected changes. Additionally, we plan to introduce renegotiation mechanisms for tackling such unexpected events that require the agreement to be adapted. Finally, we also consider studying different proposal based on learning agents' preferences and annealer mechanisms that further improve the performance.

Acknowledgements This work is supported by TIN2012-36586-C03-01, TIN2011-27652-C03-01, and TIN2009-13839-C03-01 projects of the Spanish government, the FPU grant AP2008-00600 awarded to Víctor Sánchez-Anguix, and the New Governance Models for Next Generation Infrastructures project with NGI grant number 04.17. We would also like to thank anonymous reviewers for their valuable feedback.

References

1. Plan de ahorro de eficiencia energética de los edificios públicos de la generalitat. *Diari oficial de la Comunitat Valenciana*, (6800):18038–18044, june 2012.
2. S. Abras, S. Pesty, S. Ploix, and M. Jacomino. An anticipation mechanism for power management in a smart home using multi-agent systems. In *Information and Communication Technologies: From Theory to Applications*, pages 1–6. IEEE, 2008.
3. M. Klein, P. Faratin, H. Sayama, and Y. Bar-Yam. Negotiating complex contracts. *Group Decision and Negotiation*, 12:111–125, 2003.
4. P. Kuo, R. Schroeder, S. Mahaffey, and R. Bollinger. Optimization of operating room allocation using linear programming techniques. *J Am Coll Surg*, 197(6):889–895, 2003.
5. O. J. Mengshoel and D. E. Goldberg. The crowding approach to niching in genetic algorithms. *Evolutionary Computation*, 16(3):315–354, 2008.
6. L. Perez-Lombard, J. Ortiz, and C. Pout. A review on buildings energy consumption information. *Energy and buildings*, 40(3):394–398, 2008.
7. V. Sanchez-Anguix, S. Valero, V. Julian, V. Botti, and A. Garcia-Fornes. Evolutionary-aided negotiation model for bilateral bargaining in ambient intelligence domains with complex utility functions. *Information Sciences*, 222:25 – 46, 2013.
8. B. Sareni and L. Krahenbuhl. Fitness sharing and niching methods revisited. *IEEE transactions on Evolutionary Computation*, 2(3):97–106, 1998.
9. A. Schumann, N. Wilson, and M. Burillo. Learning user preferences to maximise occupant comfort in office buildings. *Trends in Applied Intelligent Systems*, pages 681–690, 2010.
10. A. Sorici, O. Boissier, G. Picard, and A. Santi. Exploiting the jacamo framework for realising an adaptive room governance application. In *Proc. DSM'11, TMC'11, AGERE!'11, AOOPES'11, NEAT'11, & VMIL'11*, pages 239–242. ACM, 2011.