Technical Report CS14-13

Optimizing Energy Efficiency in Buildings using Multi-Objective Genetic Algorithms

Dibyajyoti Mukherjee

Submitted to the Faculty of
The Department of Computer Science and
The Department of Environmental Science

Computer Science Reader: Dr. Robert Roos
Environmental Studies Reader: Dr. Shaunna Barnhart

Allegheny College
2014

I hereby recognize and pledge to fulfill my responsibilities as defined in the Honor Code, and to maintain the integrity of both myself and the college community as a whole.

_____

Dibyajyoti Mukherjee

Contents

List of Figures **iv**

1 Introduction **1**
1.1 Importance of Energy Efficiency 1
1.2 Optimization techniques 2
1.3 Goals of the Project 4

2 Genetic Algorithms **5**
2.1 Genetic Operators 6
2.2 Multi Objective Genetic Algorithms 7
2.3 NSGA-II 7

3 Related Work **9**
3.1 Objective Functions 9
3.2 Building variables 10
3.3 Implementation 11

4 Implementing the System **13**
4.1 Formulating the system 13
4.2 Fitness Evaluation using *EnergyPlus* 14
4.3 Extending ECJ 15

5 Case Study: Alden Hall **17**
5.1 Modeling the building 17
5.2 Results 17
5.3 Discussion 17

6 Conclusion **18**
6.1 Summary of Results 18
6.2 Threats to validity 18
6.3 Future Work 18

Bibliography **19**

List of Figures

1.1 Buildings Share of U.S. Primary Energy Consumption Uses [10] 2
1.2 Representing solutions as chromosome [15] 3
1.3 Crossover and Mutation operators [15] 3
1.4 Pareto Optimal Solution Curve [6] 4

3.1 Object Oriented Framework used by Wang et al. [17] 11
3.2 Optimization system used by Magnier et al. [11] 12

Chapter 1

Introduction

1.1 Importance of Energy Efficiency

Energy usage has been continuously increasing since the dawn of the industrial age. In the last 40 years, energy usage globally has risen by approximately 70 percent according to a 2010 World Resources Institute Report. Even more recently, the 2013 International Energy Outlook report projects a 56 percent increase in energy consumption in the next 30 years [1]. Fossil fuels supply about 80 percent of the world's energy needs with coal and oil together accounting for the vast majority of the number [16]. Rising energy use leads to an increase in greenhouse gas emissions from fossil fuels which further leads to an increase in chances of climate change. The 2007 IPCC report on climate change states that fossil fuels are responsible for 85 percent of human generated CO_2 emissions. This figure is stated to increase by another 62 percent by 2030 from 2002 levels.

Energy efficiency, i.e., "delivering the same (or more) services for less energy" [8], is one of the quickest and cheapest ways to increase the amount of energy available for use. The European Council for an Energy Efficient Economy describes energy efficiency as the 'cornerstone of a sustainable society' [9].A large share of the energy supply has to come from renewable energy sources such as wind and solar power in order to comply with international treaties such as the Kyoto Protocol. However, with increasing energy demand, the development of renewables needs to be supplemented with energy efficiency.

From an environmental perspective, an obvious benefit to energy efficiency is the reduction in the amount of greenhouse gas emissions. Reduction in greenhouse gas emissions can help with improving urban air quality, reducing acid rain, and reducing eutrophication (i.e. an increase in the concentration of nutrients in water that promotes excessive algae growth) [9]. There are economic benefits to energy efficiency as well. According to the Department of Energy's Energy Efficiency and Renewable Energy program, Americans saved $7 billion on residential energy bills in 2004 from energy saving measures and by building energy efficient homes [10]. Increased energy efficiency contributes to energy security by making a country less dependent on imported non-renewable energy sources and thereby making it more competitive in the globalized world.

The environmental impact of buildings are significant. In developed countries, buildings accoun for 70 percent of electricity consumption and 30 percent of greenhouse gas

emission during their operational phase [5]. In addition, other phases such as construction of the building, transportation and extraction of raw materials use significant amounts of energy as well [5]. In the United States, buildings consume about 70 percent of the total electricity and about 39 percent of all energy consumption [18]. Energy efficiency in buildings can thus play an important role in making the building 'green'. Some such measures are structural and can only be included in newly constructed buildings; many others can be incorporated during building refurbishment. While there are software tools available to simulate the effects and impacts of a particular design, tools for optimizing the design are not readily available [17] [15].

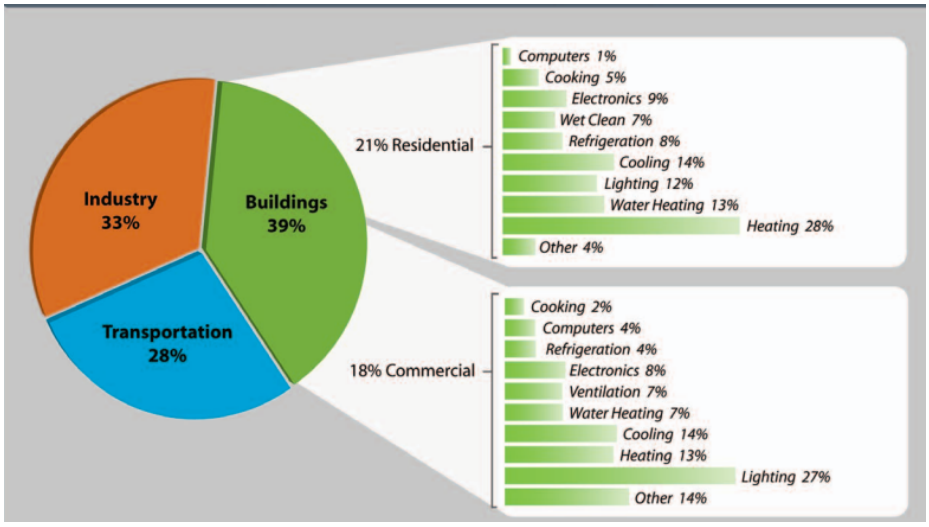

Figure 1.1: Buildings Share of U.S. Primary Energy Consumption Uses [10]

1.2 Optimization techniques

Solving an optimization problem allows us to find at least one solution that minimizes or maximizes a particular criterion. This criterion is represented by an objective or a fitness function that depends on variable parameters (both continuous and discrete) that describe the solutions [15]. Optimizations can be both single criterion or multi criteria. There are three main methods for optimization: enumerative, calculus based, and random [15]. Enumerative methods go through every single solution in the search space in order to find the optimal one. While this method is simple, it is extremely inefficient especially when it comes to problems such as building optimization due to the huge number of possible solutions. Calculus based methods use a rigorous mathematical expression of the objective function. The main limitation to this method, besides having to know an explicit mathematical expression (that sometimes has to be continuous), is that it can find a local optimum in the neighborhood without going through the global search space. Random methods, as the name suggests, use random evaluation of solutions and are often built to emulate other phenomena.

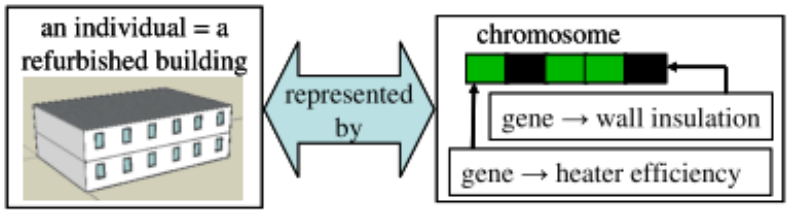

Figure 1.2: Representing solutions as chromosome [15]

Genetic algorithms (GAs) are a form of random optimization methods that seek to mimic the process of evolution in order to select an optimal solution. Strings of either real numbers or bits (binary digits, 0 and 1) are used analogous to a gene in order to represent the parameters (Fig.1.2) [15] [6]. Multiple such sub-strings are then concatenated to form the genotype. An initial population of these genotypes is generated randomly. The algorithm consists of three main functions: selection, mutation and crossover (Fig.1.3). During selection, the fitness of an individual string is evaluated using the parameter values it represents. The fittest of the strings are then crossed-over, i.e., allowed to mate and produce progeny by combining substrings of random length from pairs of genotypes. Mutation is allowed by occasional changing the values of a string position of a newly created progeny. After a number of generations of the process, the parameter values represented by the genotypes hopefully converge towards optimal solution values [6].

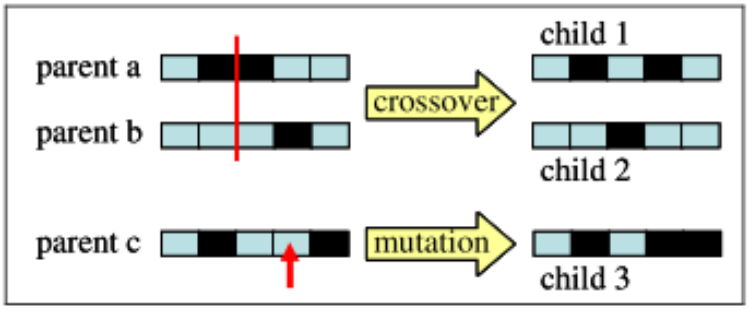

Figure 1.3: Crossover and Mutation operators [15]

The general advantages of using genetic algorithms for optimization problems are all relevant in the case of energy optimization for buildings. GAs do not require a knowledge of the mathematical structure of the problem. A GA search is not limited to a local optimum [15]. Further, multi-objective GAs such as the non-dominated sorting genetic algorithm (NSGA-II) produce Pareto optimal solutions. A solution is Pareto optimal (or non-dominated) if a decrease in one objective cannot happen without an increase in at least one other objective [7][15]. For a problem with two objectives (such as energy consumption and cost), the result is a curve of Pareto solutions instead of just one solution. This allows us to obtain a whole set of solutions from which we can then choose (Fig.1.4).

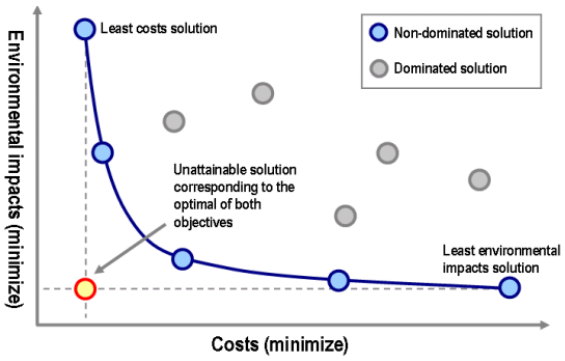

Figure 1.4: Pareto Optimal Solution Curve [6]

1.3 Goals of the Project

The aim of this proposed project is to use a genetic algorithm-based approach to create a system for optimizing energy efficiency in a building. The system will take in parameters needed to model the building design and output a set of Pareto optimal solutions. This will be a multi-objective optimization problem with two objectives—minimizing energy usage and minimizing associated cost. A multi-objective genetic algorithm, NSGA-II, will be used along with a building energy simulation program, EnergyPlus, for fitness evaluation. The next chapter provides a more detailed overview of genetic algorithms in general and NSGA-II in particular. The following chapter provides a comparative discussion of five studies specific to building optimization using genetic algorithms. Next, I will discuss the implementation details of this project and present a case study utilizing it. Finally, the conclusion will provide a summary of the goals achieved, describe threats to the validity of this project and list ways in which future work can extend the project.

Chapter 2

Genetic Algorithms

The The genetic algorithm (GA) is an adaptive heuristic based randomized search algorithm that is inspired by evolutionary processes such as natural selection and genetics. GAs were invented by John Holland in 1975 and are mainly used for optimization and search problems. A GA initially consists of a population of candidate solutions (called individuals) that are usually randomly generated. Each individual contains properties (akin to chromosomes) that can be modified by the genetic operators. Chromosomes were originally encoded as strings of bits (0s or 1s) by Holland. However, many other encodings have since been used. These include :

Permutation: The chromosome represents a sequence in a string of numbers.

Value: The chromosome is represented as a sequence of values that could be anything from real numbers to characters to user defined objects.

Tree: The chromosome consists of a tree of some objects representing functions or commands in a programming language.[]

A GA has to specify an objective or fitness function that calculates a numerical value for each individual solution that determines how well the particular solution solves the problem. The goal of the GA is, then, is to find solutions that minimize or maximize the fitness value for a given problem. The evolution loop of the GA consists of many generations. In each generation, the fitness of each individual in the population is calculated. The selection operator selects parents from the generation based on the fitness scores. The cross-over and mutation operators generate children solutions from the parents that are then used for the next generation. These operators are described in detail in next section. Some GAs include elitism i.e. some selected individuals from the parent generation is kept in the next generation. The GA stops when the stopping criteria is met. Usual stopping criteria include a generation limit (i.e. stop when the GA has run for a fixed number of generations) or a fitness limit i.e. an individual in the current generation has a fitness value better than or equal to a specified value.

2.1 Genetic Operators

Holland's original GA had four main operators—selection, crossover, mutation, and inversion. Inversion is not widely used as a operator anymore. The others are described below:

Selection

The selection operator chooses the individual genomes that will undergo cross-over. The most popular selection methods are:

- Roulette Wheel : Each genome is assigned a fitness value and the probability of selection of a genome is proportional to its fitness.
- Tournament : A tournament consists of a predetermined number of randomly chosen individuals. The entire solution space is divided into tournaments and the individual with the best fitness in each genome is chosen.
- Best : This selects the best available individuals as parents based on fitness value
- Random : This selects parents randomly.

Crossover

The crossover operator is used to produce children solution from the parents selected by the selection operator. It is analogous to biological reproduction. The main types of crossover are:

- One Point : A crossover point is selected on both parents. All data beyond that point is swapped in both parents producing two children.
- Two Point : Two crossover points are selected on both parents. All data within the two points is swapped producing two children.
- Uniform : Individual genes are compared between two parents and swapped with a fixed probability.

Mutation

The mutation operator is used in the evolution process to randomly change the offspring produced through crossover. It is analogous to biological mutation and alters one or more gene values from in a chromosome from its initial state. Mutation is needed to ensure that not all solutions fall within the local optimum of the problem. Mutation occurs during evolution according to a user defined probability. Usually a low value is used as otherwise the algorithm will turn into a random search algorithm. Some popular kinds of mutation are :

- Flip Bit : This operator takes the chosen gene and inverses the bits (1 to 0 and 0 to 1). This is only applicable for binary genes.
- Boundary : This operator replaces the value of the chosen gene with either the upper or lower bound for that gene (chosen randomly). This mutation operator can only be used for integer and float genes.

- Uniform : The uniform operator replaces the value of a gene with a random value that is generated between user defined bounds.

2.2 Multi Objective Genetic Algorithms

GAs are not limited to simply one objective function. Many real world optimization problems require two or more objectives that are conflicting. Thus, trade offs need to be made among the objectives. Multi-objective genetic algorithms (MOGAs) minimize multiple objective functions subject to a set of constraints. Formally, MOGAs can be depicted as:

$$min(f_1 x), f_2(x), f_3(x), \ldots, f_k(x)) \tag{2.1}$$

where k is the number of objectives, and $x \in X$, X being the set of chromosomes (or decision variables) that is defined by constraints. It is not possible to find a single solution that optimizes all objectives simultaneously. Therefore, we instead find a set of Pareto optimal or non-dominated solutions. As defined in 1, a solution is non-dominated (or Pareto optimal) if if a decrease in one objective cannot happen without an increase in at least one other objective [7][15]. Mathematically, for k objectives, a solution x_1 is said to dominate x_2 if

$$f_i(x_1) \leq f_i(x_2) \forall i \in 1, 2, \ldots, k \tag{2.2}$$

2.3 NSGA-II

The Non-dominated sorting Genetic Algorithm was designed by Kalyanmoy Deb et al. and published in 2002. Deb et al. use NSGA-II on a number of case studies and find that it finds a much better "spread of solutions and better convergence near the true Pareto-optimal front as compared to other GAs such as the SPEA. The main features of NSGA are:

NSGA-II has a fast computational complexity of $O(MN^2)$ where M is the number of objective functions and N is the population size. Other multi-objective genetic algorithms such as the strength Pareto Evolutionary Algorithm (SPEA) have a complexity of $O(MN^3)$ [7].

NSGA-II incorporates elitism which has been shown to improve performance of a GA significantly [7].

Diversity of solutions in a population is ensured using a crowd comparison operator which alleviates the need for an external sharing parameter.

The two distinguishing features of the NSGA-II algorithm are its fast non-dominated sorting approach and diversity preservation using a crowd comparison operator. NSGA-II sorts individuals into fronts (or ranks) based on their domination level.A non-dominated front (or Pareto front) is a set of solutions that are non-dominated (or Pareto optimal) [7]. For

each front, individuals in preceding fronts are ignored and the non-dominated front is calculated among the remaining solutions [7].A naive approach for sorting the solutions in fronts based on their domination level would involve comparing a solution with every other solution resulting in $O(MN^2)$ comparisons for each front and $O(MN^3)$ comparisons for generating all N fronts (with M denoting the number of objective functions and NN the population size). Deb et al. use a novel approach that involves, for a solution x, the calculation of the domination count, the number of solutions that dominate a particular solution and a set of solutions that x itself dominates [7]. The process requires $O(MN^2)$ comparisons. The trade-off here is that the naive approach requires $O(N)$ space while Deb et al. 's approach has $O(N^2)$ space complexity.

The crowd comparison operator is used for preserving diversity among individuals in a population. It is a faster alternative to the sharing variable approach to diversity preservation that is used by other multi-objective genetic algorithms such as the original NSGA [7]. First, the algorithm estimates the density of solutions surrounding a solution by calculating Euclidean distances for each objective [7]. The crowd-distance value is the sum of the individual values for each objective. The crowd comparison operator is then used to select a diverse non-dominated (or Pareto optimal) front [7]. This operator prefers the individual with the lower rank when comparing two individuals in different ranks. Between two individuals in the same front, the one located in a lesser crowded region gets precedence [7].

As with many other GAs, the first step in the algorithm is generating a random parent population that is then sorted based on non-domination [7]. For all generations besides the first, the main loop first creates a population of $2N$ by combining the population of the previous and current generations to ensure elitism. The combined population is sorted by non-domination rank. The new population is created by adding individuals in the order of their ranks. To select exactly N individuals, the crowd comparison operator is used in the last rank to be included in the new population [7].

For real coded chromosomes, Deb et al. use a simulated binary crossover (SBX) and polynomial mutation operators instead of other traditional ones [7]. Other studies have empirically proven that these operators deliver better results when real encoded individuals [2]. Simulated binary crossover, as the name suggests, simulates the effect of a crossover operation in a binary encoded GA on a real encoded GA. In doing so, it overcomes many of the shortcomings of other crossover operators for real coded GAs [2] and provides a search power similar to that of a crossover operation in a binary coded GA. Search power, here, is defined in terms of "the problitiy of creating an arbitrary child solution from a given pair of parent solutions" [2]. SBX uses a probability distribution function along with a user defined function called the peakedness of the distributions [3]. Analogous to SBX, the polynomial mutation operator simulates the children distribution of bit-flip mutation in binary encoded GAs on real valued chromosomes [3]. This again includes a user defined parameter that helps to generate children that are far off from the parent in real encoded GAs [3].

Chapter 3

Related Work

Genetic algorithms have been used to solve optimization problems for new buildings as well as for refurbished buildings. While some studies have focused on energy efficiency, others have included other factors such as material selection and structural design [5] [4]. In this chapter, I will analyze five different studies involving building optimization using genetic algorithms as outlined in Table 3.1 These five studies have significant differences in their goals and depth, thereby providing a broad overview of the topic. I will mainly concentrate on how the authors formulated the objective functions, what variables they used in the genetic algorithm, and how they implemented their system.

Table 3.1: Overview of the five studies reviewed in this chapter

Author	Year	Study Focus
Magnier et al.	2010	Thermal comfort and energy usage optimization using GA and neural networks
Milajic et al.	2013	Methodology for green building design using Multi-objective GA
Pejicic et al.	2012	Optimal energy efficient building design using GA and Tabu search
Pernodet et al.	2009	GAs for building refurbishment optimization
Wang et al.	2005	Object Orient Framework for building energy usage optimization

3.1 Objective Functions

Wang et al. present a case study that uses their multi-objective framework to design a single story office building in Montreal, Canada [17]. The objective functions they use are the total life cycle cost and life cycle environmental impact for a green building design. They use exergy, i.e., the maximum theoretical work that can be done by a system with respect to its environment, as the indicator for environmental impacts [17]. Besides the exergy consumed by the building during its life cycle, they also consider the exergy required to remove or recover the wastes produced in the different life cycle phases of the building such

as greenhouse gases [17]. For the cost objective, they consider the initial construction cost, the operation cost, and the pre-operation cost including resource extraction, transportation, etc. [17].

Milajec et al. use similar objective functions as Wang et al. i.e. life cycle cost and life cycle environmental impact. While they do include both cost of construction and of operation, they do not specify whether they include pre-construction cost as well. Their life cycle environmental impact assessment cost includes the cumulative exergy (like Wang et al.) but not the exergy required for waste recovery or removal. The Pernodet et al. study uses some similar objective functions as well. The main difference is that instead of two they use three objective functions —yearly energy consumption of the building in KwH/m^2, investment cost related to the refurbishment of the building, and an economic global cost defined as the "sum of initial investment cost, the yearly energy cost and the yearly maintenance cost" [15]. Their study looks at optimizing the refurbishment of a school building in France.

Unlike the above studies, Magnier et al. and Pejicic et al. do not use cost as an objective function. Magnier et al. focus on thermal comfort as an objective function aside from energy consumption. Pejicic et al. use a case study to demonstrate multi objective optimization using a model single floor commercial building. Their goal is the optimization of insulation materials and orientation angle of a given building.

3.2 Building variables

Pejicic et al consider three building variables in their studies —building orientation, and inner and outer insulation materials from a set of ten different materials. Pernodet et al. [15] focus on the building envelope parameters for their objective function. These include thermal transmittance of the walls and windows, glazing ratio and solar factor, air tightness of building envelope, and artificial lighting power. Similarly, Milajec et al. mainly consider structural variables as well incluidng wall, floor, and roof type and layers, building shape, orientation and structure. The Wang et al. study is much more in depth as they consider the most variables out of any of the above studies. Further, many of their variables are composed of other sub-variables. For instance, they have two different wall types and each wall type has a sub-variable called insulation among other ones. Insulation is a discrete variable that can take many different values, each corresponding to a different insulation material. Each wall type has a specific set of insulation materials to choose from. Besides wall type, Wang et al. look at floor types, roof type, and building shape. Each of these has its own set of sub-variables making the Wang study significantly more complex than the others. While the above studies consider only variables related to the building envelope, Magnier et al. use HVAC system-related variables in addition to the building envelope ones. They use a total of 20 different variables including heating and cooling set points, relative humidity set point, supply air flow rates, thermostat delays, window sizes, thickness of wall etc. The window sizes are separate for each wall and the HVAC system related variables are considered separately for two different seasons in a year.

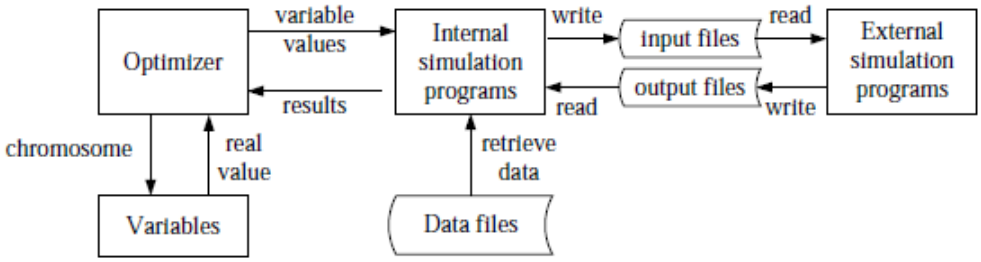

Figure 3.1: Object Oriented Framework used by Wang et al. [17]

3.3 Implementation

Wang et al. [17] present an extensible object oriented framework implemented in C++ and Fortran (Fig. 3.1). The object oriented nature of the framework presented makes it easily extensible and promotes code reuse. The framework supports both continuous and discrete variables and implements algorithms to solve both unconstrained and constrained single objective optimization problems, but only unconstrained multi-objective optimization problems. Constrained, as the name suggests, limits the range of values for a variable to certain range as specified by the user while unconstrained does not. The framework integrates with the commercially available ASHRAE Toolkit energy simulator program and can be extended to integrate with *EnergyPlus*. The authors use the framework for a case study with impressive results and present a number of ways in which the project can be extended. While this framework can certainly be very useful as a starting base for my proposed project, it is not readily available as a open source project.

Unlike Wang, other studies focus less on the software framework and more on the method of approach of completing the optimization process. Indeed, Pejicic et al. state that a "solving methodology using evolutionary algoritms" as their objective. A major difference in implentation between the Wang et al. study and the others is the energy simulation software used —both Milajec et al. and Pejicic et al. use EnergyPlus while Magnier et al. use TRYNSYS. While TRYNSYS is widely used in industry (CITE), EnergyPlus is newer and cross-platform, and is freely available from the Department of Energy. In addition, it supports modelling a building through Sketchup through a free plugin. The ASHRAE Toolkit is older than EnergyPlus and does not support as many thermal modelling options either. Pernodet et al. do not use thermal simuation at all. Instead they estimate energy usage mathematically. This method is simple though it might be inaccurate for more complex buildings.

One of the main drawbacks of using an energy simulation program is that they are time consuming. Wang et al. report that a complete optimization run for their case study took about 70 hours. EnergyPlus and TRYNSYS would be even slower since their thermal simulation model is more complex. To counter this problem, Magnier et al. use an artificial neural netowrk (ANN) for fast evaluation. They use a multilayer feed-forward ANN to first mimic the behavior of the base building model, and then use this ANN inside the GA for

fast evaluation of individuals (Fig. 3.2). This technique is very efficient and reduces the computational time associated with each evaluation to almost negligible while maintaining a good accuracy [11]. The main limitation to this method is that the ANN needs to be trained before it can be used for the evaluations. The training period can take a relatively large amount of time. For instance, the training period in the Magnier et al. study was around 3 weeks. The advantage, however, is that the once the ANN is trained, it can be used for multiple runs of the system. A different approach would be to use a distributed model of computing by dividing the task of evaluating the fitness of an individual among a cluster of computers instead of a single one. While this method has not been used for optimizing building design in particular, it has been used for other similar multi-objective optimizaton problems.

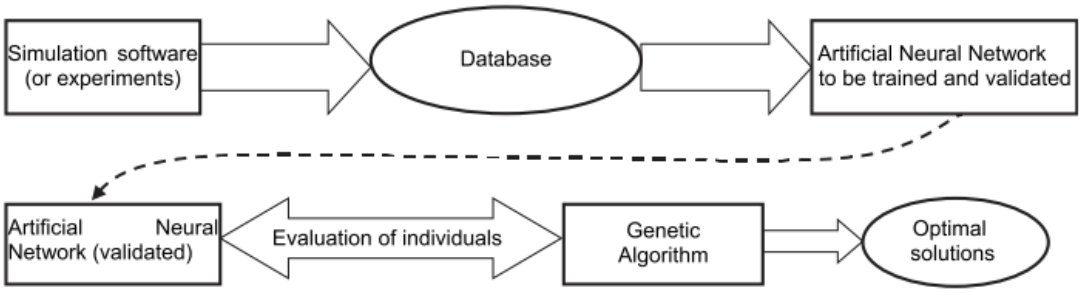

Figure 3.2: Optimization system used by Magnier et al. [11]

Chapter 4

Implementing the System

The system is implemented using version 7 of the Java programming language using version 21 of the Evolutionary Computation for Java (ECJ) framework. *EnergyPlus* version 8.1 is used for simulating energy usage of the buildings. The implementation was done on Mac OSX 10.9 and Debian Linux version 7.0 running on the Google Compute Engine cloud platform.

4.1 Formulating the system

The two objective functions are:

- Minimizing the total energy usage. This includes the total electricity usage for a year as well as energy used for heating.
- Minimizing the associated life cycle cost. This includes the costs associated with making construction (i.e. material and labor costs) as well as the cost of operation.

Variable name	Type	Unit
Window glazing Material	Discrete	N/A
R Value of Insulation Material	Discrete	m^2K/W

Table 4.1: Building envelope variables

There are 5 different building variables that comprise the chromosome for an individual building in the system. These can be classified into two main categories—building envelope related, and HVAC system related. Building envelope system related variable include window glazing material, window sizes, and insulation material used (Table. 4.1). Glazing type is chosen from a set from the set of five commonly used types as specified by the Energy Information Administration (Table. 4.2). All insulation materials that come bundled with *EnergyPlus* are supported. The costs are determined using data from the Energy Information Administration [13]. The HVAC system related variables include the HVAC system type (Table. 4.4), and heating and cooling set points (Table. 4.3).

The building variables were chosen on the basis of a number of factors. First, they had to be supported by *EnergyPlus* as otherwise their effect cannot be quantized during

Glazing Type
Gas-fills
Heat absorbing
Low emissivity coatings
Reflexive coatings
Spectrally selective coatings

Table 4.2: Glazing type for windows

fitness evaluation. Next, the variables had to serve a practical purpose. For instance, while building orientation and shape can influence energy efficiency, it is unlikely that they can be changed (unless one is designing a new building from scratch). Similarly, *EnergyPlus* supports district heating as a HVAC system type. However, it is not feasible to implement such as system since they are implemented on a community scale. Finally, the variables had to have been used in a recent related work and have had a non-negligible impact in the results of those studies. Magnier et al. [11] have used heating and cooling set points as well as window glazing materials while Pejičić et al. [14] have used insulation materials for walls. While the type of HVAC system has not been used in any of these studies, my hypothesis is that switching to more efficient HVAC systems will have a significant change in the overall energy use of a building.

Variable name	Type	Unit
Heating Set Point	Continuous	°C
Cooling Set Point	Continuous	°C
HVAC System Type	Discrete	N/A

Table 4.3: HVAC System variables

Name
Furnace
Heat Pump
Central Boiler
Packaged Unit

Table 4.4: HVAC System Types

4.2 Fitness Evaluation using *EnergyPlus*

Energy simulation for the building is a central part of the fitness evaluation. *EnergyPlus* is a building energy simulation program developed by the US Department of Energy (DOE) that can be used for energy analysis and building load simulation. According to the product website [12]:

> Based on a user's description of a building from the perspective of the building's physical make-up and associated mechanical and other systems, *EnergyPlus* calculates heating and cooling loads necessary to maintain thermal control set points, conditions throughout a secondary HVAC system and coil loads, and the energy consumption of primary plant equipment.

EnergyPlus version 8.0 is used in the project for running annual energy simulation for each individual building in a population. The simulation is run using a time step of 15 minutes as this is the minimum that the *EnergyPlus* documentation recommends. *EnergyPlus*, by default, is a command line based program that uses a text based IDF file for input and can produce output in many different formats including HTML and CSV. A number of plugins are also available that add a graphical user interface.

Initially the building to be optimized is modeled in OpenStudio, a free plugin for *EnergyPlus* with a graphical user interface. The resulting text based IDF file is modified by ECJ during runtime and used by *EnergyPlus* to run the simulations. The output is produced in the form of a comma separated value (CSV) file that lists the annual as well as monthly energy and electricity consumption in Joules. This information is used by ECJ to determine the fitness value for the individual building. A single building simulation in *EnergyPlus* can take between 15 seconds to more than a minute. The difference is a result of the complexity of the building design being used. A sample simulation building with 3 floors divided into 3 zones with a fully configured HVAC system as well as solar panels took 70 seconds to run. Using this time as an average, an optimization using a population of 20 and 100 generations would take approximately 39 hours to run.

4.3 Extending ECJ

The ECJ implementation of the NSGA-II algorithm as well as that of the selection, mutation, and crossover operations are used in the system. Individuals are represented using ECJ's `DoubleVectorIndividual` as arrays of `doubles`. The population size, number of generations, crossover and mutation types can be specified using a text based parameter file (4.5). The range for each building variable (gene) can also be specified using the parameter file.

Parameter Name	Values
Population size	Integer Value (usually between 50 and 100)
Number of generations	Integer Value
Crossover type	One point, Two point, Simulated binary, or Uniform
Mutation type	Uniform, Gaussian, or Polynomial
Selection type	Tournament, Roulette, Random, or Best

Table 4.5: Genetic Algorithm Parameters supported by the system

A custom fitness evaluation method was created by implementing the `ec.simple.SimpleProblemForm` interface and extending the `ec.Problem` class. The

`evaluate` method in this class reads in an `Individual` among other parameters. The genome for the `Individual` is used to modify the input file for *EnergyPlus*. The modified input file is then used to run the annual energy usage simulation for the building. The output from the simulation i.e. the total electricity and the total energy from natural gas is used to assign the fitness value for each objective. The energy usage objective is assigned the total energy usage for the year. The associated cost is calculated by multiplying the electricity and natural gas usage numbers by their respective average rates for the year 2013 for the state of Pennsylvania. The rates are assigned from the data provided by the Energy Information Administration [13].

The fitness evaluation is carried out in parallel using ECJ's in-built distributed master slave evaluation model from the `ec.eval` package. This model consists of one master node along with a number of slave nodes. The master node handles the evolutionary process and sends the individuals to be evaluated at the various slave nodes. As with the rest of the system, the parameters for this module is also specified in a parameter file. These parameters include the IP address for the master node, the number of jobs to be shipped to a slave at once, the number of jobs per slave among others as shown in Table 4.6. Currently, the system is set up on f1-micro instances of the Google Compute Engine cloud platform. There are 12 slave nodes and 1 master node all running Debian Linux version 7.0. This can easily be changed to any other network by just modifying the IP addresses in the parameters file.

Parameter Name
Master IP address
Master port number
Job size for shipping
Number of jobs per slave
Slave name

Table 4.6: Parameters for distributed evaluation

Chapter 5

Case Study: Alden Hall

5.1 Modeling the building

Alden Hall is building on the Allegheny College campus that houses that Computer Science and Geology departments. The building was built in 19XX and then renovated in 19XX. This two storey building is 119”8’ feet long and 58’6” feet wide on the outside with XX windows. The building is oriented ?? direction. Sketchup 2013 and OpenStudio v.1.2 were used to model the building and then convert it to the EnergyPlus readable IDF format.The match photo feature of Sketchup was used to model the building exterior. Four photographs from each of the four corners of the building were used. The interior of the building was modeled with the help of floor plans provided by the Allegheny College Physical Plant. The Open Studio plugin for Sketchup was used to transform the Sketchup Model into thermal zones using the secondary school template. This template provided automatic defaults for the people usage loads as well as a schedule for how the loads differed on weekdays, weekends, and holidays.

5.2 Results

5.3 Discussion

Chapter 6

Conclusion

6.1 Summary of Results

6.2 Threats to validity

- Inaccurate thermal modeling of case-study building
- Energy consumption for building as reported by EnergyPlus not verified with actual consumption records
- Weather file for Meadville PA not found in the NREL database. Using Erie/Pittsburgh instead.
- Cost is based on energy rates for only one year.

6.3 Future Work

- Expanding building variables.
- Life cycle study -> considering life cycle energy impact of the building materials
- Modularizing the system. Ability for user to choose variables etc.
- Improving the cost function to include rates of change in natural gas/electricity/labor prices. Amortized payment over a period of time.
- Closer integration with OpenStudio using their Ruby user scripts.

Bibliography

[1] U.S Energy Information Administration. International energy outlook 2013. Technical Report DOE/EIA-0484, US Department of Energy, July 2013.

[2] Ram Bhusan Agrawal, Kalyanmoy Deb, and Ram Bhushan Agrawal. Simulated binary crossover for continuous search space. 1994.

[3] Sunith Bandaru, Rupesh Tulshyan, and Kalyanmoy Deb. Modified sbx and adaptive mutation for real world single objective optimization. In *Evolutionary Computation (CEC), 2011 IEEE Congress on*, pages 1335–1342. IEEE, 2011.

[4] Negin Behboudi, Fouad Butt, Abdolreza Abhari, and Victoria Street. Automatic Building Design with Genetic Algorithms and Artificial Neural Networks(WIP). *Proceedings of the 2012 Symposium on Simulation for Architecture and Urban Design*, 3:7, March 2012.

[5] Daniel Castro-Lacouture, Jorge a. Sefair, Laura Flórez, and Andrés L. Medaglia. Optimization model for the selection of materials using a LEED-based green building rating system in Colombia. *Building and Environment*, 44(6):1162–1170, June 2009.

[6] David A Coley and Stefan Schukat. Low-energy design: combining computer-based optimisation and human judgement. *Building and Environment*, 37(12):1241–1247, December 2002.

[7] Kalyanmoy Deb, Associate Member, Amrit Pratap, Sameer Agarwal, and T Meyarivan. A Fast and Elitist Multiobjective Genetic Algorithm : NSGA-II. 6(2):182–197, 2002.

[8] US Environmental Protection Energy. Clean energy.

[9] European Council for an Energy Efficient Society. Why energy efficiency? `http://www.eceee.org/why_energy_efficiency`, 2013. [Online; accessed 1-May-2013].

[10] World Resources Institute. Power surge: Energy use and emissions continue to rise. `http://www.wri.org/publication/content/8601`, 2010. [Online; accessed 1-May-2013].

[11] Laurent Magnier and Fariborz Haghighat. Multiobjective optimization of building design using TRNSYS simulations, genetic algorithm, and Artificial Neural Network. *Building and Environment*, 45(3):739–746, March 2010.

[12] U.S. Department of Energy. Building Technologies Ofice: EnergyPlus energy simulation software. http://apps1.eere.energy.gov/buildings/energyplus/, 2013. [Online; accessed 1-May-2013].

[13] US Department of Energy. Energy information administration. http://http://www.eia.gov/, 2013. [Online; accessed 10-December-2013].

[14] Goran Pejičić, Miloš Vrbanac, and Milenko Vukadinović. Optimal Energy Efficient Building Design Using Improved Evolutionary Algorithm. *Archives for Technical Sciences*, 4(7):43–48, November 2012.

[15] Fanny Pernodet, H Lahmidi, and P Michel. Use of Genetic Algorithms for Optimization of Building Refurbishment. In *Eleventh International IBPSA Conference*, pages 188–195, Glasgow, 2009.

[16] A. Adegbululgbe J. Fenhann I. Konstantinaviciute W. Moomaw H.B. Nimir B. Schlamadinger J. Torres-MartÃ■nez C. Turner Y. Uchiyama S.J.V. Vuori N. Wamukonya X. Zhang R.E.H. Sims, R.N. Schock. Energy supply. In P.R. Bosch R. Dave L.A. Meyer B. Metz, O.R. Davidson, editor, *Climate Change 2007: Mitigation. Contribution of Working Group III to the Fourth Assessment Report of the Intergovernmental Panel on Climate Change*. Cambridge University Press, Cambridge, United Kingdom, 2007.

[17] Weimin Wang, Hugues Rivard, and Radu Zmeureanu. An object-oriented framework for simulation-based green building design optimization with genetic algorithms. *Advanced Engineering Informatics*, 19(1):5–23, January 2005.

[18] Weimin Wang, Radu Zmeureanu, and Hugues Rivard. Applying multi-objective genetic algorithms in green building design optimization. *Building and Environment*, 40(11):1512–1525, November 2005.