Passive Solar Building Design Using Genetic Programming

by

Mohammad Mahdi Oraei Gholami

A thesis submitted to the School of Graduate Studies in Partial fulfillment of the requirements for the degree of Master of Science

Department of Computer Science
Brock University, St Catharines, Ontario

Abstract

Passive solar building design is the process of designing a building while considering sunlight exposure for receiving heat in winter and rejecting heat in summer. The main goal of a passive solar building design is to remove or reduce the need of mechanical and electrical systems for cooling and heating, and therefore saving energy costs and reducing environmental impact. This research will use evolutionary computation to design passive solar buildings. Evolutionary design is used in many research projects to build 3D models for structures automatically. In this research, we use a mixture of split grammar and string-rewriting for generating new 3D structures. To evaluate energy costs, the EnergyPlus system is used. This is a comprehensive building energy simulation system, which will be used alongside the genetic programming system. In addition, genetic programming will also consider other design and geometry characteristics of the building as search objectives, for example, window placement, building shape, size, and complexity. In passive solar designs, reducing energy that is needed for cooling and heating are two objectives of interest. Experiments show that smaller buildings with no windows and skylights are the most energy efficient models. Window heat gain is another objective used to encourage models to have windows. In addition, window and volume based objectives are tried. To examine the impact of environment on designs, experiments are run on five different geographic locations. Also, both single floor models and multi-floor models are examined in this research. According to the experiments, solutions from the experiments were consistent with respect to materials, sizes, and appearance, and satisfied problem constraints in all instances.

Acknowledgements

I would like to thank the following individuals who supported me and did not leave me alone in all aspects of my research:

- Brian Ross for his guidance, financial support, and excellent supervision.
- Adrian Harrington and Maryam Baniasadi for sharing their entire source code with me.
- Cale Fairchild for his network, clusters computing, and Linux expertise.
- Tania, my wife, and my parents, Maryam and Jalil for their financial and spiritual support.

M.M.O

Contents

List of Tables

List of Figures

Chapter 1

Introduction

In a passive solar building design each part of a building, such as the walls, floors, roofs, and windows, are designed to collect heat in winter and reject heat in summer without using a mechanical system. Mechanical systems use gas and electricity. In addition, most of them make noise. Therefore, not using mechanical systems makes a building energy efficient and quiet. The main reason that there is a lot of research in passive design is that it does not need too much effort to build a passive solar building [5]. Passive solar buildings have few moving parts of mechanical systems compared to other design applications [2]. There are three different ways that designers and architects can benefit from using computers in designing buildings and structures: using computer aided design tools, interactive evolutionary systems, and automated evolutionary systems. Computer aided design tools help designers and architects in analyzing the structure, simulating, and measuring the performance. Interactive evolutionary systems helps a designer to a explore design search space, while his or her interaction with the computer is necessary for guiding evolutionary process toward the final design. Pure evolutionary systems evolve solutions without interaction with the designer. Designers need to define objective(s) and evolutionary process will lead individuals toward fitter solutions that meet constraints.

1.1 Goals and Motivations

In this research, we implement a genetic programming (GP) application which can design buildings automatically considering energy performance efficiency. Our goal is to develop an automatic system for designing buildings using passive solar techniques to reduce or minimize the need of using mechanical devices for cooling and heating. Our main contribution will be creating 3D models of buildings, considering material for each surface and efficient design of windows, doors, and overhangs. We also consider the geographic location. Environmental impacts on the building, such as rain, snow, temperature, humidity, and wind are also included. We do not consider cost of materials, aesthetics, decoration, and floor plans. As we will define objectives for fitness of our designs, our evolutionary approach does not need any human effort during evolution. We use split and string-rewriting grammars to manipulate and develop 3D models. For evaluation, we use energy simulation software called EnergyPlus, to give us information about building performance such as energy needed for cooling and heating, heat loss, and heat gain. Then a multi-objective ranking method will be used to rank each building based on the objectives.

Architects and designers can use evolutionary architectural designs for further exploration of designs. Our system can give them ideas about new possible designs. Then, they can refine or modify the part of designs which are not feasible or logical to them.

1.2 Thesis Structure

The thesis structure is as followed. In Chapter 2, we will give a detailed background material related to our work, which are evolutionary design, genetic programming, multi-objective optimization, shape grammar, L-system, split grammar, and energy efficiency. Chapter 3 provides a literature review of evolutionary design and 3D modeling, and evolutionary design and energy efficient architecture. Chapter 4 provides detailed information about our methodology and GP system. Chapter 5 examines different energy-based objectives used in the next chapters. Chapter 6 shows the impact of geographical location in material selection, building sizes, and window placements. In Chapter 7, we study multi-floor building models. Chapter 8 compares our work with related research. Finally, Chapter 9 summarize the thesis and how to extend this work.

Chapter 2

Background

2.1 Evolutionary design

Evolutionary computation is a branch of artificial intelligence inspired by natural Darwinian evolution [7]. In evolutionary computation, the problem is defined in terms of a search space. Each point in the search space is a potential solution. In all evolutionary techniques, there is an initialization phase. Mostly in initialization phase, a collection of solutions called a population is selected randomly. Then a heuristic method is used to navigate solutions in the search space toward better solutions. In fact, the heuristic method looks at previously visited solutions to decide where to look next. The whole navigation process of previously visited solutions toward new solutions is an evolutionary process. Genetic algorithm, genetic programming, evolution strategies, and evolutionary programming are different branches of evolutionary computation. They differ in how they represent the search space, the way how they navigate the search space. Evolutionary computation can be used in many application areas [7, 21].

Evolutionary design is the application of evolutionary computation to design forms, structures, and models. For many years architects and designers used computer aided design applications to simulate, analyze and measure the performance of their designs [5]. Architects and designers relied on these applications to test their new ideas and designs before they build real prototypes from them. Today, computers are also being used for generating new and innovative designs by means of evolutionary computation. The main goal is not to replace architects and designers with computers, as many applications that operate interactively with designer or architects. Generating a perfect design is not the goal. Automated evolutionary design may give ideas to designers for designing new models and finding new aspects of design while considering limitations such as materials, boundaries, cost, etc.. This type of evolutionary design is called open-ended [25]. There is another type of evolutionary design called parameterization [25]. Here, evolutionary design is used to refine or optimize an existing design or structure. In this representation, the topology of the design is pre-defined and the goal is to optimize an objective of an existing design or structure.

2.2 Genetic Programming

Genetic programming (GP) is an evolutionary computation technique [4, 27, 28, 40]. Individuals in this evolutionary process are computer programs. They are evolved based on Darwinian natural selection during generations. Each individual in the population represents a potential solution to the problem. We should keep in mind that GP, like every other evolutionary technique, does not guarantee an optimal solution. GP tries to find good solutions, and solutions will be ranked based their fitness.

GP individuals are denoted by a tree. Internal nodes are called functions and leaves are terminals. Functions have one or more children. A function set can be as simple as add and subtract, be complex specific operators. Terminals are usually data that are input to the parent node. Figure 2.1 shows a GP tree which is calculating $a + (a * b - sin(30))$.

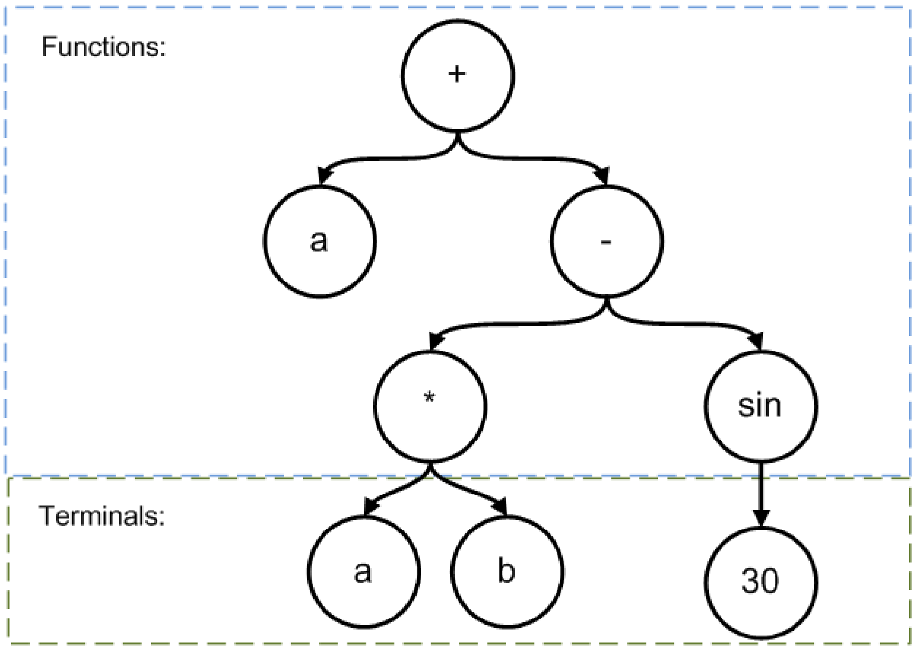


Figure 2.1: An example of a GP tree. Functions and terminals are specified by the surrounding blue and green box respectively.

Algorithm 1 Genetic Programming Pseudo Code

```
1. Initialization: load GP parameters and generate a random population of size N
using function and terminal set.
2. Evaluation: evaluate the initial population.
while Termination condition is not met do
    3. Selection: select two individuals from the population based on their fitness.
    4. Breeding: perform crossover on the selected two individuals with the proba-
    bility of $P_{crossover}$ and mutate them each with the probability of $P_{mutation}$.
    5. Repeat step 3 and 4 until (N-M) children is created where M is the elite
    number.
    6. Elite Selection: Discard the population except the best M individuals.
    7. Update the population with the new (N-E) children with M individuals from
    step 6 and then evaluate them.
end while
8. Report logs and statistics from the whole run.
```

Algorithm 1 shows a pseudo code of steps used in a GP system. In the remaining part of this section, different terms and operators in evolutionary process will be defined.

2.2.1 Initialization

Initialization is the first step shown in Algorithm 1. When initially generating individuals for the first generation, random tree generation is used. Initialization with completely random trees would result very similar trees with low diversity. Therefore Koza [27] proposed ramped half and half generation for initialization. Ramped half and half generates approximately the same number of balanced and unbalanced trees for the first generation with different depths. Balanced trees are called full trees and unbalanced trees are called grow tree. A full tree is a tree with depth d, that all nodes except nodes in depth d are functions and all nodes in depth d are terminals. A grow tree is a tree that has at most depth d and all nodes of depth 1 to d are either functions or terminals. Ramped half and half generates all full trees and grow trees with the height from $depth_{min}$ to $depth_{max}$.

2.2.2 Fitness Function

Steps 2 and 7 in Algorithm 1 are evaluation steps. A fitness function evaluates an individual to figure out how well it solves the problem of the interest. In some problems, fitness functions measure the amount of error, and zero is the best value.

2.2.3 Fitness-based Selection

Selection phase is the third step in Algorithm 1. Fitness-based selection is used between two consecutive generations for generating new offspring from individuals in the population. Tournament selection and Roulette wheel are the most famous kinds of selection. In tournament selection, a predefined number of individuals among all individuals are randomly selected, and the best individual of them is chosen. For performing crossover, two tournaments will be run for identifying the two parents. In Roulette wheel, the probability of selection an individual is proportional to its fitness value. Better individuals are more likely to be selected.

2.2.4 Crossover

Crossover is a breeding process and the breeding phase is the fourth phase in Algorithm 1. Crossover is based on inheritance in the natural evolution. Crossover is the process of creating two new offspring by selecting two random nodes from two selected individuals, and swapping their sub-trees. In Figure 2.2, two trees in the top are parents and offspring are the two trees in the bottom. Offspring will be made by swapping two branches of the parents that being marked in the figure.

2.2.5 Mutation

Mutation is another breeding process. Mutation is based on the process of mutating DNA between parents' DNA and offspring DNA. Mutation involves selecting a random node from a selected individual, and replacing it and its sub-tree with a new randomly generated sub-tree. In Figure 2.3, the parent is at the top left, the new random tree is in the top right, and the mutated offspring is at the bottom.

2.2.6 Elitism

In evolutionary computation, elitism means preserving M best individuals from last generation and injecting these M individuals to the next generation. The advantage of using elitism is to preserve best individuals of the last generation. Elitism also can have negative impacts on evolution by converging the population prematurely to a sub-optimal level, and prevent searching the rest of the search space. Elite selection is mentioned in Algorithm 1 as the 7th step in evolution process.

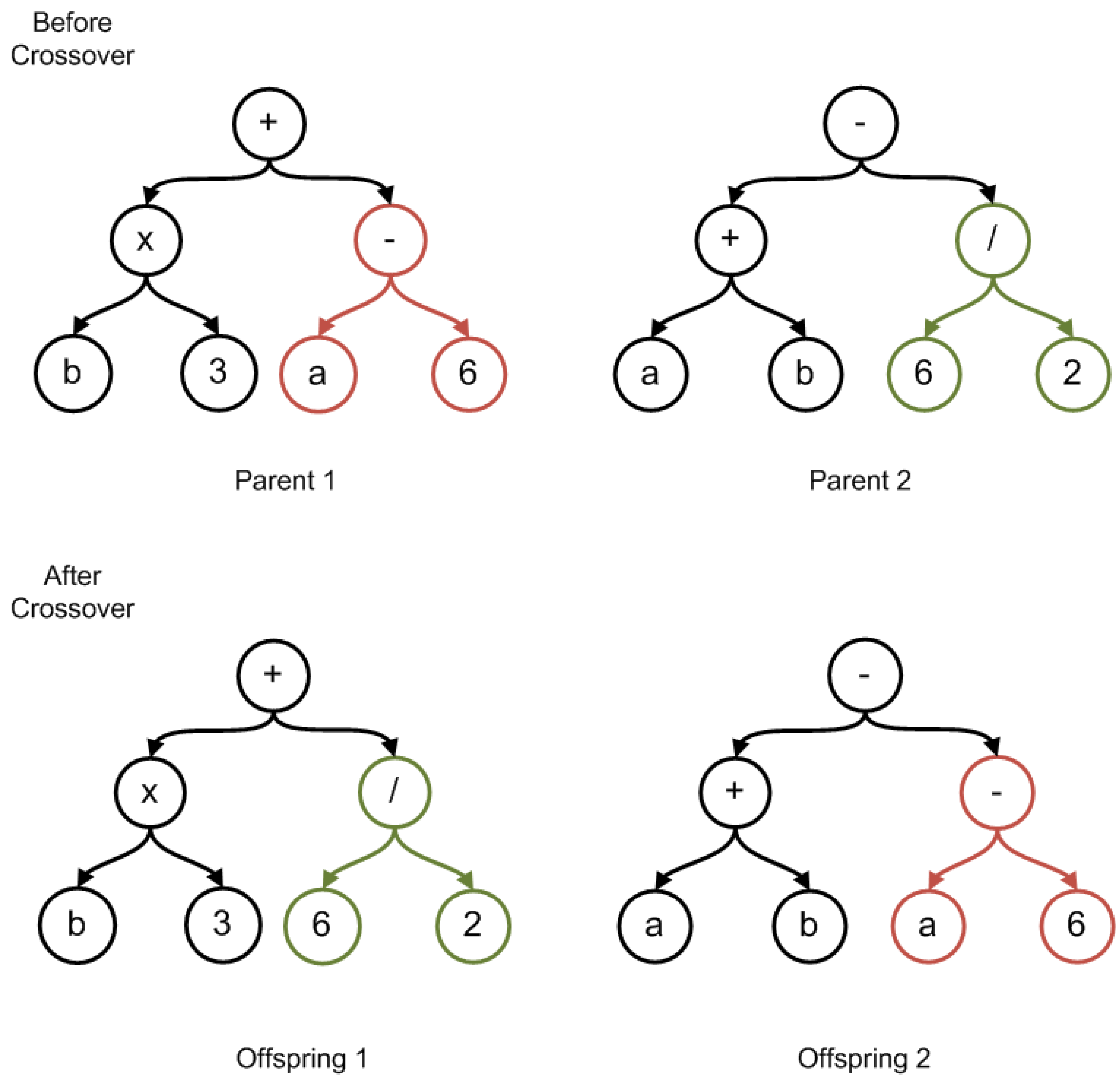


Figure 2.2: Crossover: parent 1 and parent 2 are the selected individuals for crossover. Red nodes in parent 1 will be swapped with green nodes in parent 2. Offspring 1 and 2 represent new offspring created after crossover applied on parents.

2.2.7 Ephemeral Random Constant

Ephemeral random constants (ERCs) are special kind of terminals in GP which represent random values such as 3.14159, true, 12, etc.. ERC is initialized with a random value and keeps the value during the resut of its existence. In case of mutation, another new value will replace its value.

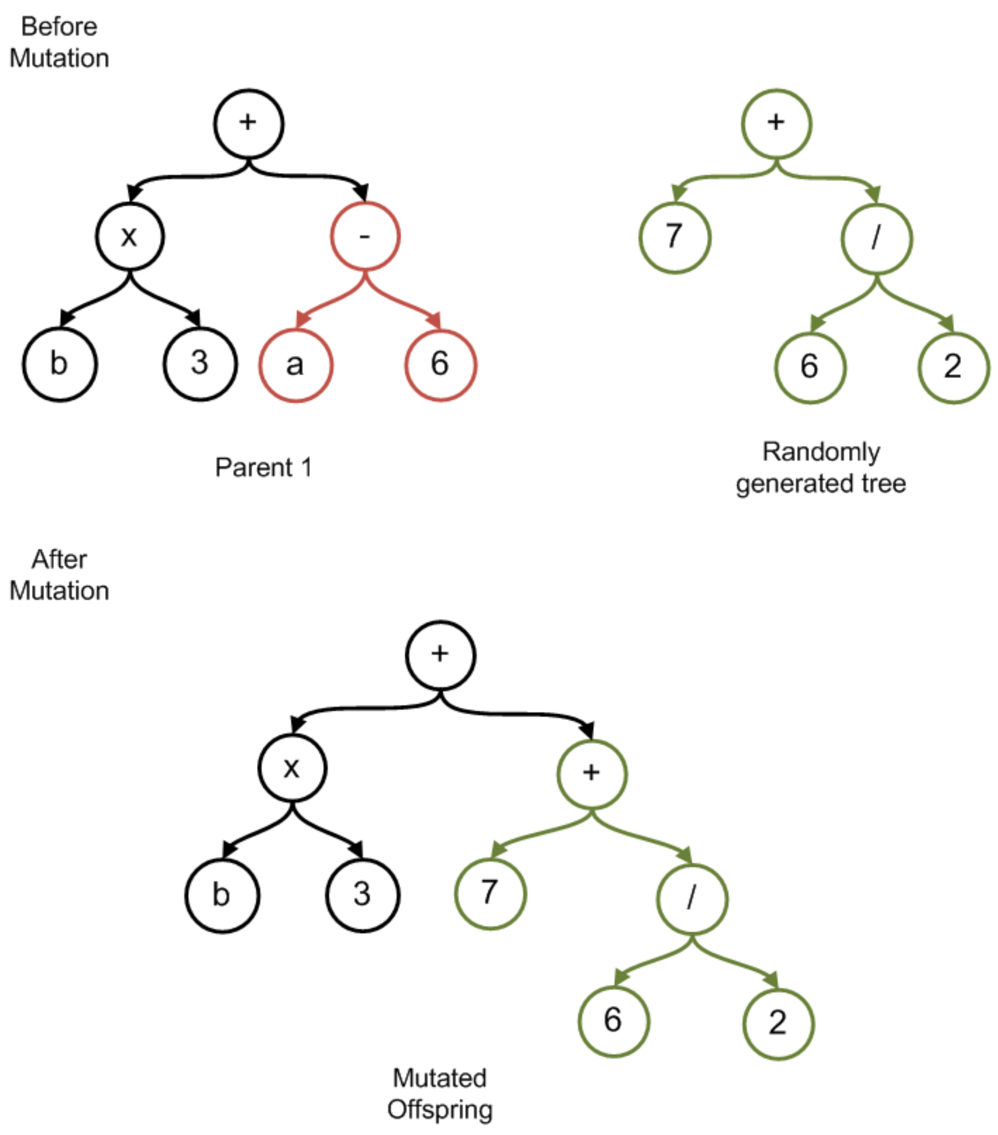


Figure 2.3: Mutation: parent 1 is the individual selected by selection method. Red nodes in parent 1 will be replaced by the randomly generated tree. Mutated offspring shows parent 1 after mutation.

2.2.8 Termination

While the termination criteria is not met, the evolutionary process continues. The termination criteria is mentioned in Algorithm 1 in the third line as a while loop. Most of the time, there are two criteria for terminating the generation: meeting the fitness criterion for a problem, or exceeding the predefined number of generations.

2.3 Multi-Objective Optimization

In many problems, there is more than one objective and these objectives need to be optimized simultaneously. Most of the time these objectives have conflicts which means improving one objective can cause poor values for other objectives. Therefore we should find a systematic way to tackle this issue. There are many techniques for solving multi-objective problems which we will now discuss [16, 20].

2.3.1 Weighted Sum

One way to represent a problem is to aggregate the values of the objectives and make a single objective from multi-objective problem. Assuming having n objectives, fitness of an individual can be calculated by the following formula:

$$fitness = \sum_{i=1}^{n} weight_i * f_i$$

where $weight_i$ is the weight of objective f_i The problem of this method is that there is no straightforward way of finding weights for objectives. In addition, weights introduce bias into the solutions found.

2.3.2 Pareto Ranking

Pareto ranking is a classic multi-objective ranking strategy which refuses combining objectives to a single objective for ranking [16]. In Pareto ranking, individuals are being evaluated on each objective and then they will be ranked based on which individual dominates others. An individual dominates another individual if it is better than the other individual in at least one objective and it is as good as the other individual in the other objectives. Having more than one individual that cannot be dominated by others is common in Pareto ranking. All these individuals get rank one. The remaining individuals that are undominated by the rest get rank 2. This process will continue until all individuals have been assigned a rank.

There are a few disadvantages in Pareto ranking. First of all, if an individual has the best score in one objective, it will be ranked 1. However, it could have very poor scores in the other objectives. These individuals considered as outliers and often are not favorable. Secondly, it is not practical to have more than 7 objectives with Pareto ranking, because most individuals will be undominated.

2.3.3 Normalized Rank-Sum

Rank-sum(RS) [6], also known as average ranking [18], is an alternative to Pareto ranking and was invented for high dimensional multi-objective problems. In RS, first we rank individuals based on the first objective. Then we rank them based on the second objective, and so on until all objectives are ranked. Then we add the ranks of the objectives for each individual to give the individual a single rank value, called the RS.

$$RS = \sum_i rank_i$$

Normalized rank-sum (NRS) normalizes each rank by dividing the rank of an individual on an objective by the maximum rank assigned to an objective in that specific objective. Therefore, if we say max_i is the maximum rank in i^{th} objective, fitness is calculated by the following formula:

$$NRS = \sum_i \frac{rank_i}{max_i}$$

Individuals are sorted descending based on NRS. In able 2.1, we have shown objectives for 6 individuals. Lower value in each objective is preferred. Then we have calculated their fitness in each ranking system. Best individual in each type is marked yellow.

Fitness	**Weighted Sum**	**Ranks**	**Pareto Rank**	**RS**	**NRS**
(33,0,125,39)	197	(3,1,6,3)	1	13	2.27
(30,24,38,18)	110	(2,3,3,2)	1	10	1.4
(0,47,43,18)	108	(1,4,4,2)	1	11	1.73
(78,62,2,0)	142	(6,6,1,1)	1	14	1.37
(43,19,20,79)	161	(4,2,2,4)	1	12	1.47
(55,55,89,80)	279	(5,5,5,5)	2	20	2.67

Table 2.1: An example of different types of ranking. Best in each ordering is highlighted.

2.4 Shape Grammar

A shape grammar was proposed by Stiny [44], and is a variation of generative grammar. A shape grammar is a context-free grammar and it is a systematic method of transforming, manipulating, and generating shapes. In Stiny's definition, a shape is a finite set of straight lines in a Cartesian coordinate. Each grammar has a set of rewriting rules for exploring the search space. Each rewriting rule has two parts: predecessor shape and successor shape. Only rules were predecessor models the current shape or part of the current shape, can be selected and applied on the current shape. Rules must be applied sequentially. In each step, a rule is selected and will be applied. New shapes can be created by applying rules. Figure 2.4 shows that rectangle EICJ is created after two rectangles ABCD and EFGH has intersected. Emerging new shapes during applying rules adds complexity to the system. Later, Stiny [43] revised his idea by adding labels to shapes. Now, only shapes that have labels are subject to manipulation.

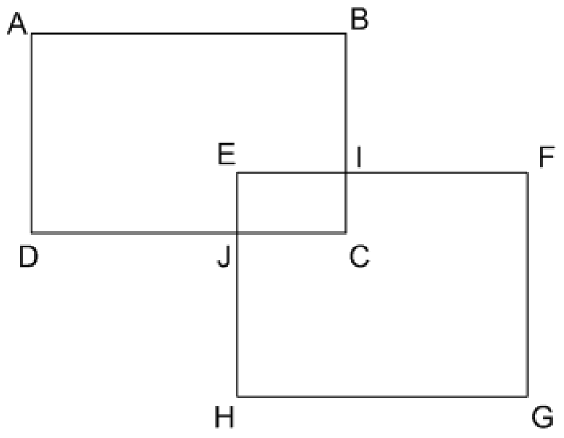


Figure 2.4: This image shows how complex can be a shape grammar. Two rectangle ABCD and EFGH have intersected and rectangle EICJ is appeared.

2.5 Split Grammar

A split grammar is a specialized kind of shape grammar [47]. In a split grammar, one problem iteratively decomposes into simpler sub-problems. After each sub-problem is solved, results will merge and will be used as a solution to the actual problem. Split grammars' rules have more restrictions than those in shape grammars. These rules are powerful enough to make model for buildings and simple enough to be performed automatically. Figure 2.5 illustrate some split grammars' rules used in this research.

A grammar generally is defined over vocabulary B by <U,+,-,F, $\leq$ >where U is the power set of B and it contains all objects in the grammar and is closed under add, and remove operations. Assuming function f $\in$ F, the grammar is closed over $f(u)\ where\ u \in U$ as well. In split grammars, vocabulary B contains all the basic shapes and U contains all the shapes that can be made by the grammar over vocabulary B with add, remove, and f ∈ F operations. In shape grammar, F contains rules of the grammar. Assuming $u, v \in U$, u "occurs" in v if there is a transformation $f \in F$ with $f(u) \leq v$.

There are two different types of rules in split grammar: split rule and conversion rule. Each split rule decomposes the basic shape into smaller shapes. Each conversion rule transforms a shape into another shape. Rules are in the form of $\alpha \rightarrow \beta$. Rule $\alpha \rightarrow \beta$ in split grammar can be performed on shape S, if and only if α occurs in S. Figure 2.5 illustrates an example of rules. Figure 2.6 shows the result after derivation of the rules on the initial shape called start in Figure 2.5.

Each shape has its own attributes such as size, orientation, etc.. After conversion or split rule is used, these attributes will be passed to their children. Attributes have different granularity and they can specify both general features and detail features of the model.

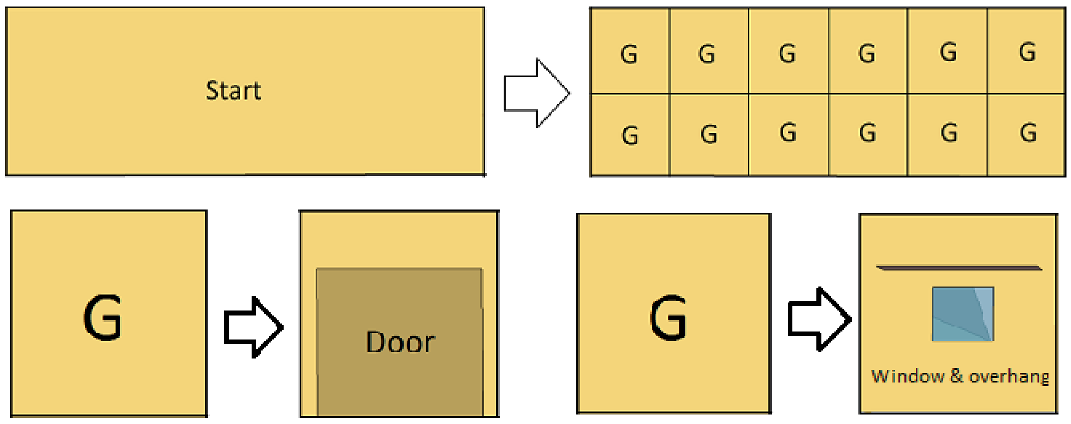


Figure 2.5: An example of rules for a split grammar. The rule at the top is a split rule and shows a wall which will be split into 12 sub-walls. The rule at bottom left is a conversion rule and shows the conversion of a sub-wall to wall and door. The rule at the bottom right is a conversion rule and convert a sub-wall to window, overhang, and wall.

2.6 Energy Efficiency

The most energy efficient buildings are called green buildings [26]. A green building is a sustainable resource-efficient building which does not have any negative impact such as waste, air pollution, water pollution, indoor pollution, storm water runoff, and noise on environment [1]. Efficient usage of non-renewable energies such as fossil

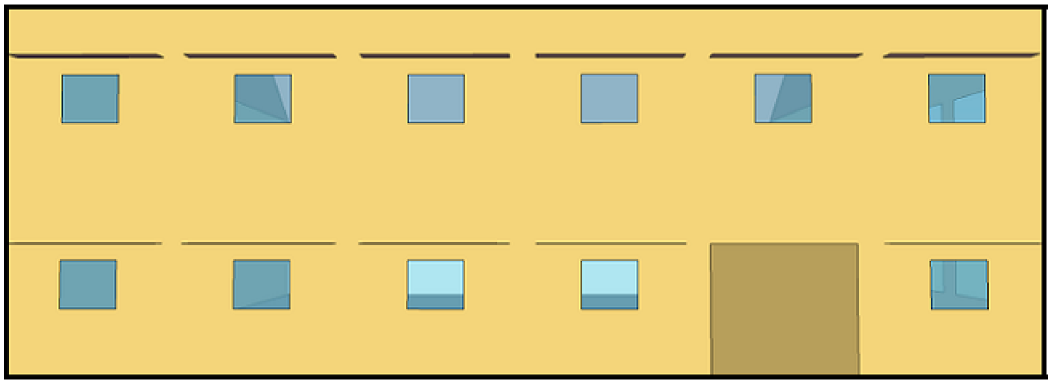


Figure 2.6: This figure shows the result of applying a split grammar on initial shape using rules.

oil, natural gas, fuels, and electricity, alongside using renewable energies such as solar energy and wind energy is a goal of energy efficiency. Reducing the cost and the amount of energy that is needed for providing services and products and reducing pollution caused by using energy is a practical result of energy efficiency. According to a USGBC report published in 2006 [1], over 70 percent of the green building research is focused on energy and atmosphere research. Therefore, in most cases when designing a green building, designers, architects, and engineers try to minimize fossil fuel energy and electricity usage to minimize environmental negative impacts. One way to minimize fossil fuel energy usage is using passive and active solar techniques for heating during winter and not overheating during summer. Warming water for home usage, electricity generation, and internal lighting are other passive solar applications.

There are many commercial and non-commercial building analysis applications for thermal simulation and analysis, energy simulation and consumption, heating ventilation and air conditioning (HVAC) consumption, daylight simulation, and energy performance. Two examples of these applications are as follows.

EnergyPlus is a free energy simulation, that considers load calculation, building and energy performance, heat and mass balance, water use, and energy flow [35]. More than 2000 local worldwide weather files are available for free use. The EnergyPlus input file can be shown by SketchUp (a 3D viewer). Accuracy and detailed simulation capabilities are its strength. EnergyPlus can be run on Windows, Mac OS, and Linux. Over 85,000 copies of EnergyPlus were downloaded since April 2001.

TRNSYS (transient system simulation program) is a commercial HVAC analysis and sizing. It considers multi-zone airflow analyses, electric power simulation, solar design, building thermal performance, and analysis of control schemes. Building descriptions, system characteristics, and connections between different systems are all ASCII input files. Weather files are supplied with program. All input files can be generated by graphical user interface with the ability to drag and drop components to make the whole model. There is a plug in for SketchUp to make 3D models and import them into TRNSYS. Having extensive libraries is one of TRNSYS strengths. However, users must provide detailed information about the building and systems to TRNSYS. Windows is the only computer platform that can run TRNSYS. TRNSYS commercial license costs $4500 and its educational license is $2250.

We will talk more about EnergyPlus in section 4.1. For more information about other applications refer to [35].

Chapter 3

Literature Review

3.1 Evolutionary Design and 3D Modeling

The followings are representative examples of research using evolutionary design in 3D modeling.

Hornby used genetic programming for designing tables and robots [25]. He used a generative representation, specifically L-systems, so elements of his code can be encapsulated and used many times. Therefore, he did not need to generate four legs for a table; instead, he only needed to generate one leg, encapsulated it as a module, and uses it four times. In addition, he used an open-ended representation. Therefore, he let his system explore the design search space to give some new potential designs to be generated.

Hornby later defines modularity, hierarchy, and regularity in generative representation [24]. Modularity encapsulates a group of elements and treats them as a unit. These units are in fact building blocks of a system. Modularity is inspired from a human problem solving methodology: divide, conquer, and merge. Modularity can benefit more in conjunction with regularity and hierarchy. Regularity is the repetition of modules and units in design. Hierarchy is the ability to use modules inside the other modules. Hornby's experience shows that using modularity, regularity, and hierarchy at the same time gives better results than any smaller combination of them or not using them at all.

O'Reilly *et al.* integrated evolutionary computing and generative growth for form exploration [39]. GENR8 uses a combination of grammatical evolution with an extension of L-systems, called Hemberg extended map L-systems (HEMLS). GENR8 concentrates on surfaces rather than materials or internal architecture. GENR8 can automatically calculate fitness value based on surfaces' size, smoothness, soft boundaries, and symmetry.

O'Neill *et al.* use grammatical evolution for designing shelters [38]. They used a plug-in for Blender 3D software to make designs. This plug-in benefits from shape grammars and grammatical evolution. In addition, this plug-in is capable of performing interactively. The main goal of their research is making elements of a design to be coherent with respect to other design elements. They developed a grammar of curves, and lists of beams that are created by some points on these curves. Fitness evaluation is done by interactive evaluation.

Byrne *et al.* [10] extended the system in [38] by considering material cost and physical constraints alongside aesthetic considerations. They implemented a fitness based system that measures stress in a structure and the amount of material. Their system tries to minimize both objectives, while not sacrificing aesthetic properties of the structure.

Coia [17] used GP and shape grammars to design conceptual building architectures. He used CityEngine for visualization. Therefore his shape grammar was limited to CityEngine's shape grammar. As Coia worked on conceptual architecture, he had not limited his program with too many constraints. For fitness evaluation, he considered objectives such as height matching, the number of unique surface normal, and the distances between surface normal. He also run experiments based on the combination of these multiple objectives.

Bergen [8] explores the generation of 3D models which are aesthetically pleasing. His system is based on L-systems. A GP system generates rules for the L-system. For evaluating individuals, functions such as standard deviation and symmetry, and model constraint functions such as surface area and volume, were used.

McDermott *et al.* [32] proposed a string-rewriting method for evolutionary architectural design. In string-rewriting grammar, each left hand side (LHS) function is rewritten by the right hand side (RHS) functions and terminals. In their work, GA is responsible to select rules and replace LHS with RHS. In addition, they present methods to improve design language influenced by the type of results have been obtained during previous experiments. They have used context free grammars for generating design models. Then rules will be chosen and applied by grammatical evolution. After obtaining results and comparing designs with desired design language, they refine their grammars by narrowing and widening the system design language. Narrowing means to add constraints to design language, and widening means let some parameters vary while they were hard-coded beforehand.

Linden [29] designed wired antenna by GA. Chromosomes identify the length of wires, their diameters, and their rotation. Also the space between each two consecutive wire is identified by chromosomes. Linden treated his multi-objective fashion problem as a single objective problem by using weighted sum. The goal of his research was to maximize voltage standing wave ratio, minimize lowest frequency used in the system, and to maximize highest side lobe level within an acceptable range.

Annicchiarico *et al.*[3] used GA to optimize the shape of a cantilever plate with circular hole. A mesh based design is used to control the boundary, material, and loads in the system. The goal of this research is to minimize the volume of the cantilever plate. Some constraints such as nodal coordinate, restriction on the shape of the elements, and stress constraint applied to attain practical results.

3.2 Evolutionary Design and Energy Efficient Architecture

Turrin et al. [46] designed a system considering large roofs structures that provides thermal and lighting comfort using passive solar techniques. They used ParaGen, which is a parametric genetic algorithm design tool. A parameterized representation was used which optimizes a pre-defined design and does not explore new designs. The result of their GA system is given as input to a commercial application, Autodesk Ecotect, for measuring daylight and thermal performance.

Malkawi et al. [30] implemented a system for defining the placement of windows, supply airs ducts, and return air ducts to maximize ventilation and thermal criteria. They only solve the problem for a single room that has two windows, two supply inlet ducts, and two return outlet ducts. A fixed size integer string is used as a genome in the genetic algorithm. Each value in the genome identifies a feature of the house, e.g. room length, height, and width. Computational fluid dynamics (CFD) is used for evaluation. The only drawback of CFD is that it is complex and it takes long to evaluate a simple room. As their program does not handle constraints directly, they only applied penalties for the designs which do not meet constraints.

Marin et al. [31] used a genetic algorithm for exploring search space of 3D designs. In their system, each individual has two chromosomes, one for topology and shape of a facade, and one for physical properties of each facade and windows on the walls. They used 3DS Max software for visualization. The authors only focused on winter comfort, so they tried to maximize sunlight exposure through the windows. In addition, the best models of their system were shown to a designer during evolution, so the designer can intervene and guide evolution. Moreover, they considered the loss coefficient of materials in the building.

Harrington [22] used genetic programming and L-systems to evolve structures. The author combined different objectives with a form filling objective. In one of his experiments, he included the position of sun at noon to maximize sun exposure in winter and to minimize it in summer. In another experiment, he considered different position of sun in winter to maximize winter sun exposure. In his last experiment, he considered shadows of obstacles as well in sun exposure calculations. In short, his structures react to different sun positions.

Caldas et al. [15] used GA to find the best window placements. The objective is to minimize the annual energy used for cooling, heating, and lighting. The weighted sum approach is used in the research. The building had a fixed size and 8 different sizes for width and length of the windows were given to the system. GA was responsible to find the best size for width and length of the windows. Four different experiments carried out to find the best placement for windows separately. Then their work was expanded to find the best placement of all windows at the same time.

Caldas et al. [12] used GA to find the best window placement for Alvaro Siza's School of Architecture at Oporto. Objectives of this research were the same as their previous work in [15] mentioned above. In this research, different tilt for roofs and roof monitors are considered, and shadings and overhangs were examined. The comparison of the most energy efficient building of the system with the actual building show that Alvaro Siza's School of Architecture has good solar performance.

Caldas et al. [14] presented a generative design system (GDS) that designs architectural models and evolves them toward more energy-efficient models using a multi-objective approach. The multi-objective technique used in GDS is Pareto ranking. The topology of the model which is the number of spaces and their adjacency, is given to the GDS. Then GDS, defines the size of the building and window placement. Finally, the model is evaluated by DOE2. The experiment of two-storey models with 4 rooms in each floor are examined here. The height of the 4 rooms of the first floor are the same, but all other sizes were defined by GDS. The objectives of this research are to minimize energy for lighting and heating in Chicago.

Caldas [13] in 2005 extends the work in [14] by replacing absolute amount of energy spent in the building with energy per unit area to normalize the output of energy simulation and analysis system. Comparing the results with [14] shows that larger rooms that have more energy consumption can have lower energy per unit area than smaller rooms. This navigates solutions toward larger models.

Caldas work [11] discusses designing sustainable energy-efficient buildings solutions. She uses a generative design system, called GENE_ ARCH, to generate and evolve a complete building design. The result is passed to DOE2 [23] to evaluate building energy performance. The goal is to locate the best windows placement and size to minimize cost for heating, cooling, and maximizing illumination. Energy performance and cost of materials are other important objectives which are considered in the experiments. Higher quality materials cost more, but they are more efficient with respect to energy use. Finding a good trade-off between thermal performance and cost is also examined.

Yu [49] used GP to model occupancy behavior of a single person office. She used historical data such as the length of the time that the office occupancy does not change, the length of time that the occupant has been in the office since arrival, etc. to predict presence or absence of the occupant. This prediction helps in efficiently usage of lights, heating, ventilation, and air conditioning systems (L-HVAC). The single objective fitness function she used is an accuracy function. Accuracy is the percentage of the correct prediction.

Bouchlaghem [9] used a semi-evolutionary technique called nonrandom complex method to design buildings which have good thermal performance. Different parameters such as thickness and thermal penetration coefficient are used in his system to define materials. Aspect ration (length/width), orientation, and ceiling height are variables which specify the size of the building. Also, window size and shading are used. The objective of his system was to minimize the degree of discomfort which is defined as the difference between comfort degree and dry resultant degree through 24 hour cycle in a winter month and a summer month.

Wright et al. [48] used GA to design the HVAC (heating, ventilating, and air conditioning) system of a south facing mid-level zone in a multi-storey building in UK. They assumed that the zone has one window on the south wall. The goal was to minimize the cost of the HVAC system, to minimize thermal discomfort during occupancy, and to minimize constraint violations of the models. Two different cost rates for electricity are used, while a flat rate for gas consumption is considered. To expedite the simulation, a three design days analysis is used: a winter design day, a summer design day, and a "swing" season day. Pareto ranking is the multi-objective technique used.

Shea et al. [42] used multi-criteria ant colony optimization to optimize lighting performance of the models. Each wall or roof is divided into squares of 1x1 meter and can use different materials. Objectives of their work include maximizing daylight, minimizing the number of hours that a specific point in the model receives direct sunlight, and minimizing the cost of the panels. In addition, conductive heat loss of the panels are also considered to show the possibility of including another objective such as thermal performance to the system. Pareto ranking is the multi-objective technique used.

Chapter 4

System Details

4.1 System overview

In this section, we will overview our GP system, how we design energy efficient buildings with GP system, and how we evaluate them. Our GP Java-based system is an extension of ECJ. We will use split grammars for designing buildings. Our GP system will define building shape and size, windows, doors, overhangs, and materials for surfaces and fenestrations.

Our energy analysis system is EnergyPlus[36]. EnergyPlus is a free application, so we can run it on many computers at a time. This is important, because designing buildings and analyzing them is a time consuming job. In addition, EnergyPlus is very comprehensive application which considers solar energy, heat ventilation air conditioning (HVAC), building materials, different, window type, piping, lights, window placement, and overhang placement. EnergyPlus considers everything that is needed for analyzing and calculating energy performance. In its simulation and analysis, geographical location of the building and the impact of the environment are important. The EnergyPlus weather file includes geographical information such as the geographical location of the building and the weather history of the location for a year. The EnergyPlus weather file includes temperature, humidity, atmospheric pressure, extraterrestrial radiation, wind direction, wind speed, snow depth, and rain precipitation quantity. EnergyPlus can be integrated with Sketchup [45], which is a free professional design and visualization applications suite. Also, its input and output are ASCII files or HTML files, which are human readable. EnergyPlus has both Linux and Windows versions. As we have access to Linux based clusters, we benefit from this service for running multiple programs concurrently. This speeded up our research. After giving the 3D model to EnergyPlus, it will calculate the amount of energy which is needed for cooling and heating, and heat gain and loss of each window.

4.2 System Execution

Figure 4.1 overviews the steps used in the execution of our system. First GP parameters and modeling parameters loaded into the system. These parameters includes generation number, population size, tree max size, crossover probability, mutation probability, and design grammar parameters such as minimum and maximum size for the height, width, and length of the building. Then evolution begins by generating random GP trees. Then individuals will be translated into input data file (IDF), which is understandable by EnergyPlus. EnergyPlus starts simulation by reading a generated IDF file and weather file, analyzing the building, and writing a report on different aspects of the simulation. As there is likely no perfect model, our system stops evolution after specific number of generations. After ranking individuals based on their objectives, the evaluation completion condition will be checked. If this generation was not the last generation, reproduction operations are applied, and new individuals are produced. Then the process goes back to the GP tree translation to IDF file step. If this generation was the last generation, statistics and best models from the last generation will be generated.

4.3 Diversity

In evolutionary design, diversity of the population is very important, because diversity prevents evolution to prematurely converge. This lets the evolutionary system search more of the search space. Flack [19] created a method to increase the diversity of the population. In evolution, individuals that have the same fitness vector will be considered as identical models and will be penalized. This penalty will be added to their rank by a predefined penalty value. As it is rare to have different individuals with the same score, checking that two GP trees are the same or not is ignored in this strategy.

4.4 GP Language

First, GP identifies width, length, and height of the building. Next, GP language uses a split grammar similar to Wonka [47]. Every split grammar divides a complex

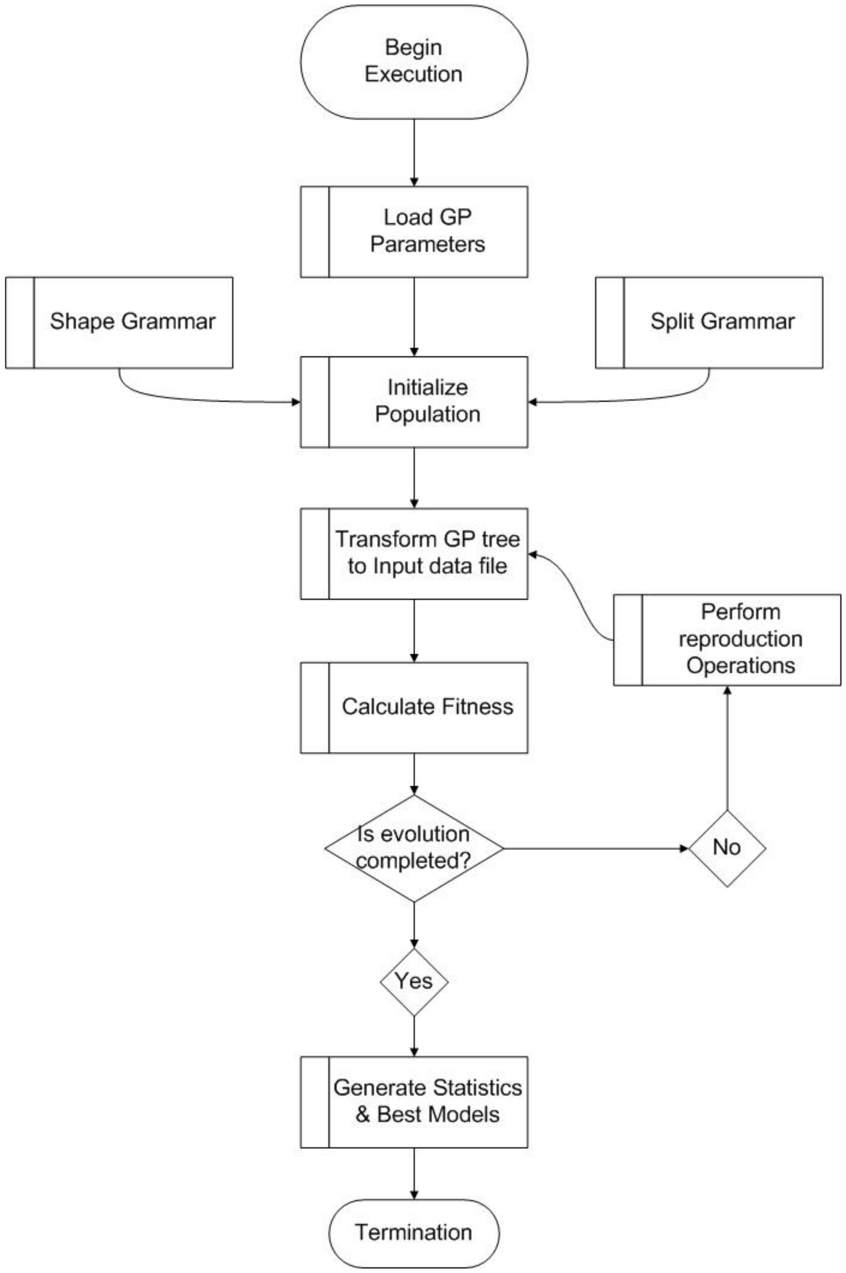


Figure 4.1: System Execution

problem into smaller sub-problems. Then after solving sub-problems, it merges the result of each sub-problem to solve the original problem. In our system, walls and roofs are divided into smaller pieces. Windows, doors, and overhangs will be placed on these smaller pieces. Note that overhangs cannot be placed on roofs. GP also uses a string-rewriting grammar similar to [32]. After a split rule is carried out, each portion uses string-rewriting to solve the sub-problem.

Discrimination between different elements of the design was required. Therefore,

Symbol	Representation	Description
R	Root	Top level tree structure
D	Double	Double value between 0 and 1 inclusive.
I	Integer	Integer value
W	Window	Window node
D	Door	Door node
FF	First floor	First floor or main floor of the building
F	Floor	Floors other than the main floor
R2	Roof	Roof of the building with skylights.
S	Shape	Shape and size of the building
G	Grid	Each facade can have a grid. Grid splits a facade into sub-facade. In each sub-facade, only one window or door can be placed
DG	Door-Grid	The facades that have this grid are the only facades that can have a door.

Table 4.1: GP Types

the GP language is a strongly typed language [34]. In a strongly typed language, type of each node's children and its return type are defined. This strategy enforces GP tree to have a specific structure. Table 4.1 shows different types that are used in GP language. Types include root, double, integer, window, door, first floor, floor, roof, shape, grid, and door grid. Root node defines the tree structure. Double and integer values are used in functions to determine sizes of the walls, windows, doors, overhangs, roofs, first floors, floors, shapes, grids, and door grids.

Figure 4.2 shows a representation of the GP tree. Root defines the tree structure and has only one child: shape. Shape can be as simple as a cube defined by its width, length, and height. Each floor has different grids for each facade. Each facade will be divided into grids and each grid can have window, window with overhang, or door. Also a grid can have neither window nor door.

4.4.1 GP Terminals and Parameters

Table 4.2, 4.3, 4.4, and 4.5 represent functions and 4.6 represents terminals which are accessible to our GP system. For functions, their return type and description are provided and for terminals the description is provided.

Root of the GP tree defines its structure and the rest of the nodes define design elements. As it is mentioned in Table 4.2, Add_Root function adds a building with the shape of S. Each shape makes a single floor e.g. Add_Cube makes a cubic floor.

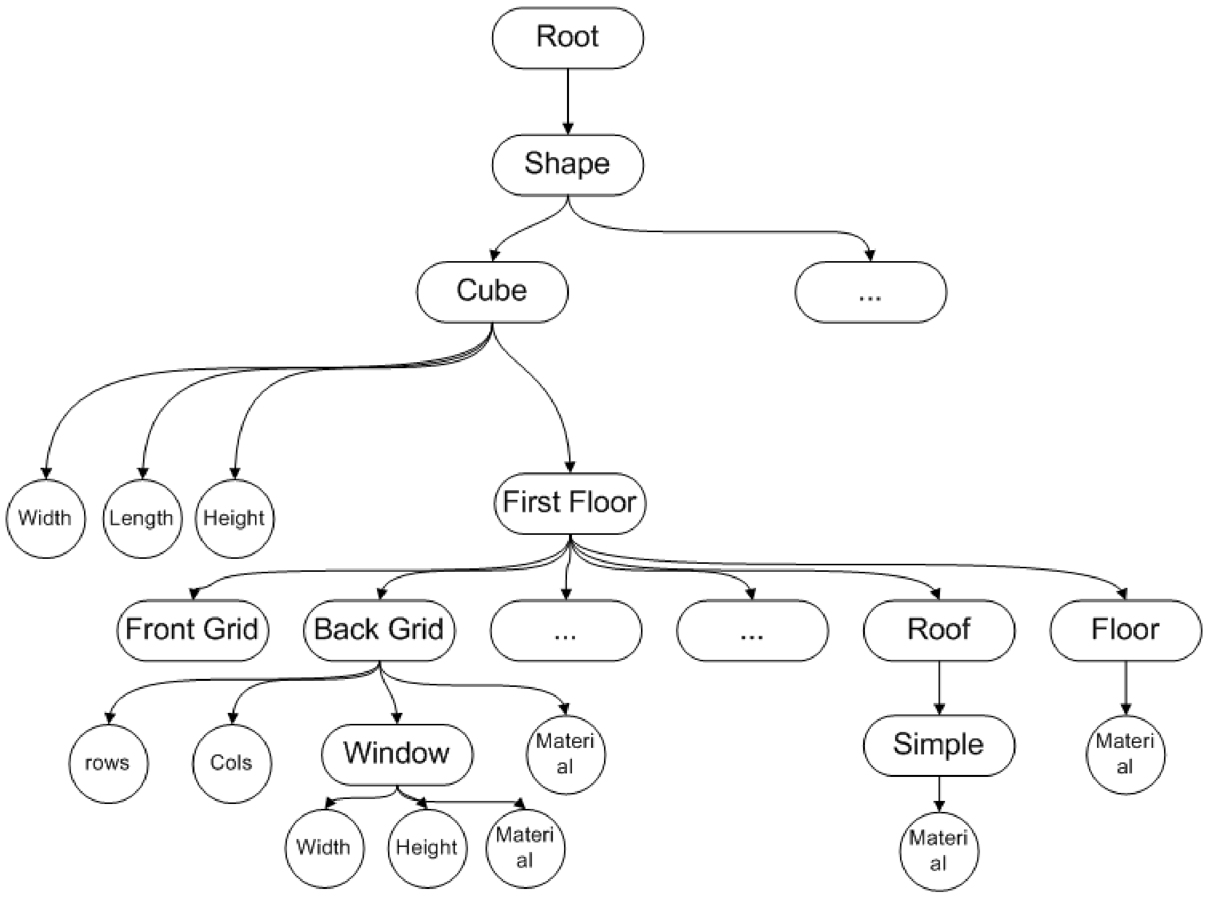


Figure 4.2: An example of GP representation

First_Floor function defines the first floor of the building. First floors will have a door grid in the front (facing to the south) and simple girds for the other sides. Add_Floor defines other floors and only can have grids without doors. Add_Grid will determine how to split a facade to sub-facades, and then which size of windows should be placed on each sub-facade. Have a facade not split into sub-facades is possible. Therefore, Add_ Empty_ Grid function is designed for this purpose.

For creating more realistic shapes and sizes, minimum size and maximum size for the height, width, and length of the building is specified. Minimum size for the length and the width is 3 meters and maximum is 20 meters. Also minimum height is 2 meters and maximum height is 4 meters. For making a building, GP returns three double values (W,L,H) which are between 0 and 1 inclusively. Width, length, and height of the building can be calculated by the following formulas:

$$Width = Width_{min} + (Width_{max} - Width_{min}) * W$$
$$Length = Length_{min} + (Lenght_{max} - Length_{min}) * L$$
$$Height = Height_{min} + (Height_{max} - Height_{min}) * H$$

Table 4.2 contains functions for making 3D models. Table 4.4 contains description of the functions that do math calculations on floating point values, and Table 4.5 contains description of the functions that performs math calculations on integers.

4.5 Example Tree Representation

In Figure 4.3 a sample 3D model generated by GP is illustrated. All the figures of 3D models from now on will be represented in the same way. Therefore, always in the top image of the model, south and east sides of the building and in the bottom image of the model, north and west sides of the building will be represented.

GP Tree corresponded to the model is also represented in Figure 4.4. As $Width_{max}$ and $Length_{max}$ are equal to 20 meters and $Width_{min}$ and $Length_{min}$ are equal to 3, according to formulas in Section 4.4.1 Shape(.99,.99) makes a building with the width and length almost 20 meters and the height of 3.66 meters. Values for doors and windows also are relative to the size of the grid and facade they are being placed on.

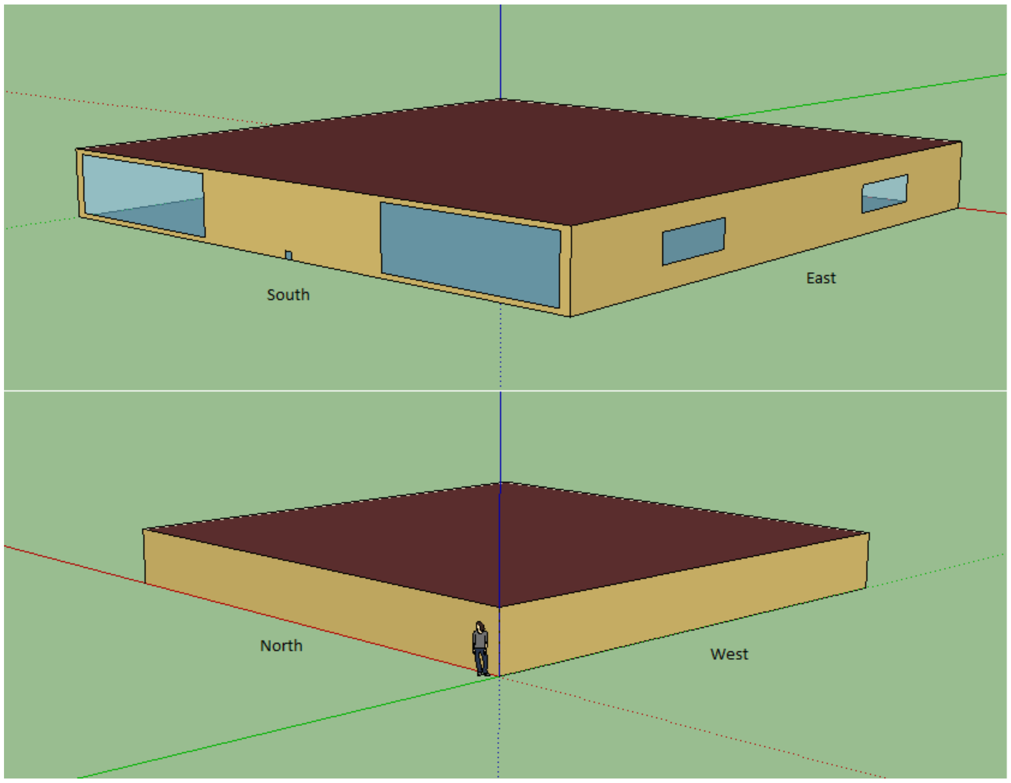


Figure 4.3: South and east side of a sample model

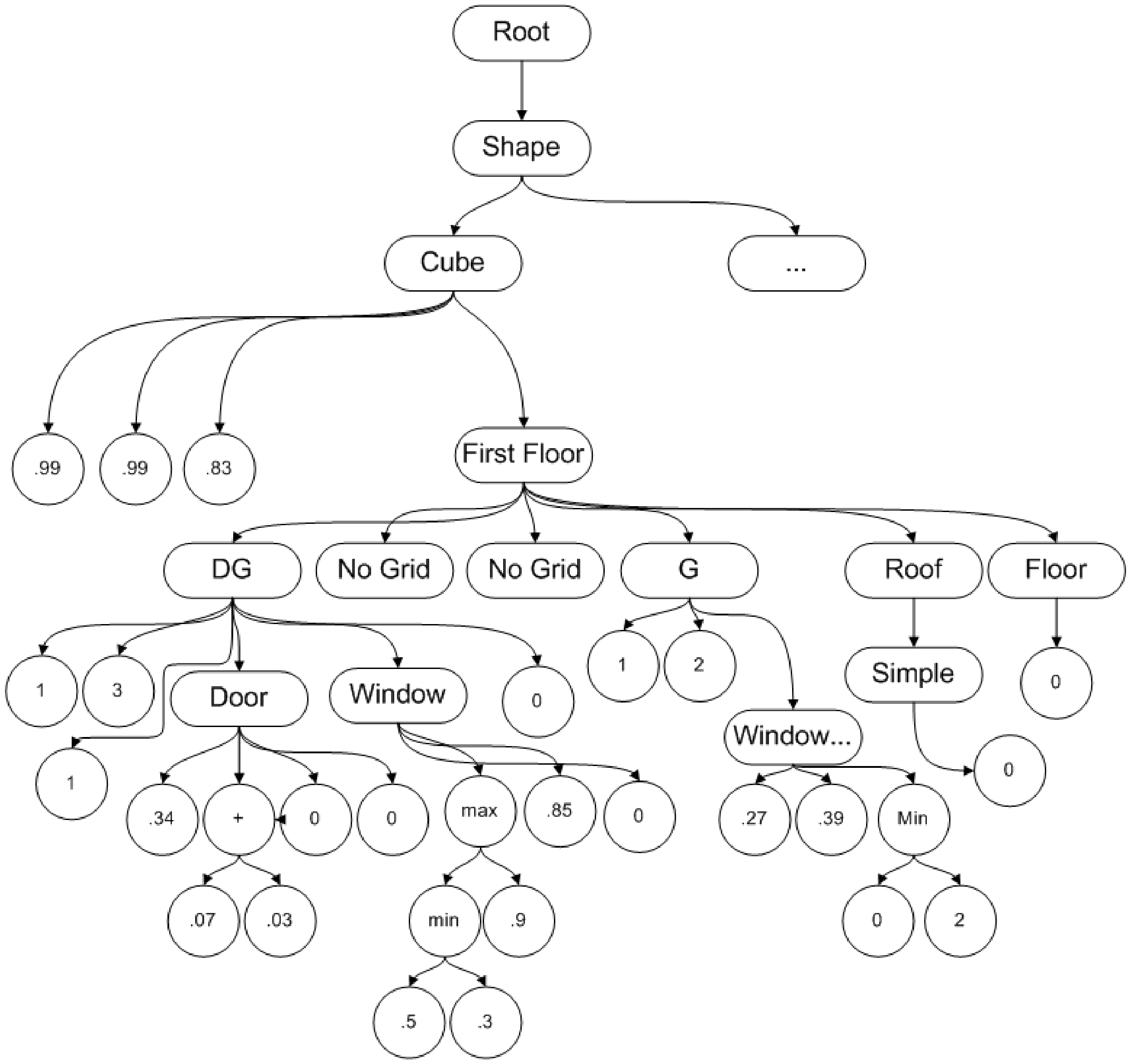


Figure 4.4: Tree representation of the sample model in Figure 4.3

4.6 Fitness Evaluation

As mentioned in Section 4.2, EnergyPlus is our energy simulation and analysis system for the building models. EnergyPlus needs two inputs to perform the simulation: input data file (IDF) and EnergyPlus weather file (EPW). IDF contains dynamic information about the 3D model, sizes, materials, equipment, etc., while EPW has static information about the location and weather condition such as wind, rain, and snow for a whole year. In this research, Toronto is chosen as location in most of the experiments. EnergyPlus input and output are shown in detail in appendix A and B respectively. After EnergyPlus simulates and analyzes the model, it writes the result in different output files. GP system reads the output files and draw out objectives useful to the system from them. In this research, multi-objective fitness evaluation is used. NRS is the multi-objective evaluation technique and it is discussed in details in Section 2.3.3.

Return Type	**Function Name**	**Description**
R	Add_Root(S)	Add a building with the shape of S.
S	Add_Cube(D,D,D,FF/F)	A simple cube as a 3D model. The first three values defines the length, width and height of the box and the last argument defines the floor.
F	Add_Floor(G,G,G,G,R2,I)	Add a simple cube shape floor. The first four arguments are grids for front, back, right, and left respectively, R2 is the roof and the last integer value defines material for the floor.
FF	First_Floor(DG,G,G,G,R2,I)	The first floor is differentiate from other floors. This function is as same as Add_ Floor function with a difference of having a door grid as the first argument.
DG	Add_Door_Grid(I,I,I,D,W,I)	Add a grid to one facade. The grid has the first argument number of rows and the second argument number of columns. The third integer determines the place that door can be installed. Forth and fifth arguments determine door and window. The last argument specifies the wall material.
G	Add_Grid(I,I,W,I)	Add a grid to one facade. The grid has the first argument number of rows and the second argument number of columns. The third argument determine window specification and the last argument defines wall material.
D	Add_Door(D,D,I,I)	Add a door to a facade. The first two arguments define the size of the door. The third argument decides to have a wooden door or glass door. The last argument specifies door's material.
W	Add_Window(D,D,I)	The first two arguments determine size of the window. The last argument specifies material.

Table 4.2: GP Modeling Functions

Return Type	**Function Name**	**Description**
W	Add_Window_Overhang (D,D,D,D,D,I)	The first two arguments determine the size of the window. The third argument shows how much the overhang goes top relative to the window which this overhang belongs to. The fourth argument determines the width and the fifth argument defines the length of the overhang. The last argument specifies window's material.
G	Add_Empty_Grid(I)	A grid which does not allow any window or door be placed on it. The argument determines wall's material.
R2	Add_Simple_Roof(I)	Returns a flat roof. The argument identifies the material for the roof.
R2	Add_Skylight(G)	Returns a skylight. The argument makes a grid at the roof and follows the similar rules that a typical grid has. The material of the roof also will be identified by the grid.
R2	Add_Gabled_Roof (I,G,G,D)	Returns a gabled roof. The first argument identifies the material for the roof. The second and third arguments make grids at the roof. The fourth argument determines its height.
R2	Add_Gabled_Roof2 (I,G,G,D)	The same as previous but differs in how it looks like.

Table 4.3: GP Modeling Functions (Continued.)

Return Type	**Function Name**	**Description**
D	Avg(D,D)	Returns the average of the two arguments.
D	Max(D,D)	Returns the maximum of the two arguments.
D	Min(D,D)	Returns the minimum of the two arguments.
D	Mul(D,D)	Returns the result of multiplication of the two arguments.
D	Div(D,D)	Returns the division of the first argument by the second argument. Returns zero if the second argument is zero.
D	IfElse(D,D,D,D)	Returns the third argument if the first argument is bigger than the second one, otherwise returns the forth argument.
D	Half(D)	Divides the argument by two and returns it.
D	HalfForward(D)	Returns $(D+1)/2$.

Table 4.4: GP Mathematical Floating Point Functions

Return Type	**Function Name**	**Description**
I	Avg(I,I)	Returns the average of the two arguments.
I	Max(I,I)	Returns the maximum of the two arguments.
I	Min(I,I)	Returns the minimum of the two arguments.
I	Mul(I,I)	Returns the result of multiplication of the two arguments.
I	Div(I,I)	Returns the division of the first argument by the second argument. Returns zero if the second argument is zero.
I	IfElse(I,I,I,I)	Returns the third argument if the first argument is bigger than the second one, otherwise returns the forth argument.
I	Increment(I)	Returns the argument plus one.
I	Decrement(I)	Returns the argument minus one.

Table 4.5: GP Mathematical Integer Functions

Terminal Name	**Description**
ERC	Returns a random constant double value and this value remains the same till the last generation unless mutation on this node happens.
Int_ ERC	Returns a random constant integer value and this value remains the same till the last generation unless mutation on this node happens.

Table 4.6: GP Terminals

Chapter 5

Basic Experiments

This chapter presents some basic experiments to investigate the impact of different 3D design elements, fitness objectives, and evolution strategies on the 3D models. Energy efficiency is a primary goal in each experiment.

5.1 Common Parameters

This section investigates the impact of different energy objectives on the evolution of the building models. Design elements are doors, windows, overhangs, and different kinds of roofs with skylights. GP parameters are listed in Table 5.1 and design parameters are listed in Table 5.2. These parameters are used in all experiments in this chapter. If any changes happen to an experiment, the changes are reported.

GP can have random behavior on different trials. Therefore, GP can give a good solution on one trial and give bad results in another trial. Therefore, a set of 10 trials is run for each experiment.

A maximum generation of 50 is used. This means that a pool of N GP-tree evolves over 50 different generations. A population of size 300 is used.

The initialization method is Koza half-and-half discussed in Section 2.2.1. The minimum and maximum depth for a grow tree is 2 and 6 respectively, and the minimum and maximum depth for a full tree is 5 and 12 respectively.

The selection method is tournament selection. A tournament of a 3 GP tree is run each time that a parent selection in breeding process is required.

Crossover and mutation are the two breeding operations. In our experiments, crossover is performed 90% or mutation is performed 10%. The depth of the new GP tree created in breeding process must be less than 17. If the depth goes over 17, the parent is copied instead.

Parameter	**Value**
Number of Runs	10
Generations	50
Population Size	300
Initialization Method	Half-and-Half
Grow Tree Max Depth	6
Grow Tree Min Depth	2
Full Tree Max Depth	12
Full Tree Min Depth	5
Tournament Size	3
Crossover Rate	90%
Mutation Rate	10%
Maximum Crossover Depth	17
Maximum Mutation Depth	17
Probability of Terminal Node Selection in Crossover/Mutation	10%
Probability of Function Node Selection in Crossover/Mutation	90%
Probability of Root Node Selection in Crossover/Mutation	0%
Elitism	2
Diversity Penalty	2

Table 5.1: GP Parameters

During the breeding process, no root node is selected for neither crossover nor mutation. Functions are selected with the probability of 90% and terminals are selected with the probability of 10%.

Elitism of size 2 is used. The best two individuals of each generation remain in the population and the rest are replaced by breeding.

Individuals with the same NRS are considered identical. One individual among them will remain with the same rank and the rest are penalized. The penalty factor is 2.

Table 5.2 shows design parameters. The minimum width and length of the building is 3 meters and the maximum width and length is 20 meters. The minimum and maximum height of the building is 2, and 4 meters respectively.

Each facade of the building will be split into smaller sub-facades. The maximum split number for the length or the width is 6 and the maximum split number for height is 2. Figure 2.6 in Section 2.5, shows a facade with 6 split in length and 2 split in height.

Table 5.3 presents different walls, roofs, and floors used in our system. A construction has layers of different materials. These constructions are used in all experiments and GP evolves models and selects materials. U-factor is the parameter that shows

Parameter	**Value**
Maximum Floor Length	20 meters
Maximum Floor Width	20 meters
Maximum Floor Height	4 meters
Minimum Floor Length	3 meters
Minimum Floor Width	3 meters
Minimum Floor Height	2 meters
Maximum Number of Rows on a Facade	2
Maximum Number of Columns on a Facade	6

Table 5.2: Design Parameters

Construction Name	**Material**	**U-Factor**
Wall_1	Wood, fiberglass quilt, and plaster board	0.516
Wall_2	Wood, plywood, insulation, gypsum	0.384
Wall_3	Gypsum, air layer with 0.157 thermal resistance, gypsum	1.978
Wall_4	Gypsum, air layer with 0.153 thermal resistance, gypsum	1.994
Wall_5	Dense brick, insulation, concrete, gypsum plaster	0.558
Roof_1	No mass with thermal resistance 0.65	1.189
Roof_2	Roof deck, fiberglass quilt, plaster board	0.314
Roof_3	Roof gravel, built up roof, insulation, wood	0.268
Floor_1	Concrete, hardwood	3.119
Floor_2	concrete, hardwood	3.314

Table 5.3: Walls, floors, and roofs material

how good a construction is in conducting heat. Constructions with smaller U-factor are not good at conducting heat and help in keeping heat inside the house. Constructions with bigger U-factor conducts heat better with outside which in some cases is not desirable.

Table 5.4 shows different type of windows, and doors used in our system. Glasses have sun heat gain coefficient(SHGC) and bigger values are better in gaining heat. For glasses, both U-factor and SHGC must be considered.

Table 5.5 shows minimum and maximum temperature parameters. EnergyPlus lets us choose different minimum and maximum values for temperature during a year. For the sake of simplicity, these parameters are constant for the whole year in all experiments. The cooling system for a building works if the temperature is over 24 °C. The heating system starts if the temperature goes below 20 °C.

Name	**Material**	**U-Factor**	**Glass SHGC**
Window_1	3 mm glass, 13 mm air, 3 mm glass	2.720	0.764
Window_2	3 mm glass, 13 mm argon, 3 mm glass	2.556	0.764
Window_3	6 mm glass, 6 mm air, 6 mm glass	3.058	0.700
Window_4	6 mm low emissivity glass, 6 mm air, 6 mm low emissivity glass	2.371	0.569
Window_5	3 mm glass	5.894	0.898
Window_6	6 mm glass	5.778	0.819
Door_1	4 mm wood	2.875	-
Door_2	3 mm wood, air, 3 mm wood	4.995	-
Door_3	Single layer 3 mm grey glass	5.894	0.716

Table 5.4: Window, and door

Parameter	**Value**
Minimum Temperature	20
Maximum Temperature	24

Table 5.5: EnergyPlus minimum and maximum temperature

Table 5.6 presents all functions which are used in creating models. All models have doors, simple flat roofs, and walls. Windows are optional and GP decides whether uses windows or not.

The design of the building is translated to input data file (IDF) which is understandable by EnergyPlus. EnergyPlus needs a weather file in simulations to consider environmental impact on the building as well. In all experiments, Toronto is chosen as the baseline geographical location. If the location of an experiment is not Toronto, the location will be mentioned. In the weather file, data such as latitude, longitude, time zone, elevation,temperature, humidity, atmospheric pressure, sun direction and illuminance, wind direction and speed, snow depth, and rain depth and quantity is in our interest. Table 5.7, and 5.8 presents sample data chosen from Toronto weather file retrieved from [37].

GP Functions	Add_Root, Add_Cube, First_Floor, Add_Door_Grid, Add_Grid, Add_Door, Add_Window, Add_Empty_Grid, Add_Simple_Roof, GP mathematical floating point and integer functions
GP Terminals	ERC, Int_ERC

Table 5.6: Design functions: these functions and terminals are used in most of the experiments of this section.

Parameter	**Value**
Latitude	43.67
Longitude	-79.63
Time Zone	-5
Elevation	173 m

Table 5.7: EnergyPlus general info from weather file

Date & Time	**TMP**	**HMDT**	**PRSS**	**S RAD**	**ILL**	**W DIR**	**W SPD**	**S DPT**	**LP DPT**	**LP QNT**
1-1-1969 1:00	-11.9	65	99220	0	0	270	4.4	1	0	0
1-1-1969 12:00	-17.2	77	98710	631	66700	248	10.8	1	0	0
1-4-1964 1:00	-6.7	55	99840	0	0	292	2.8	0	0	0
1-4-1964 12:00	.6	39	100270	728	79300	248	5	0	0	0
1-7-1981 1:00	17.6	86	100000	0	0	0	0	0	0	0
1-7-1981 12:00	21.1	82	100390	338	36800	100	4.2	0	0	0
1-10-1969 1:00	9.1	94	99380	0	0	292	2.2	0	0	0
1-10-1969 12:00	16.1	70	99770	192	20900	90	4.4	0	0	0

Table 5.8: Sample data from EnergyPlus weather file. TMP = temperature, HMDT = humidity, PRSS = atmospheric pressure, S RAD = sun normal radiation, ILL= illuminance, W DIR = wind direction, W SPD = wind speed, S DPT = snow depth, LP DPT = liquid precipitation depth, and LP QNT = liquid precipitation quantity

5.1.1 Energy for Cooling in Summer

We first examine the influence of the cooling energy objective on models. Months July, August, and September are chosen as summer. The heating and cooling system of all models are integrated. As a result, either heating or cooling can be working at a time. As we mentioned in Table 5.5, $T_{min} = 20$ and $T_{max} = 24$ degrees in Celsius is considered for all experiments. Therefore, any temperature between 20 and 24 degrees in Celsius is acceptable and neither heating system nor cooling system operate in these degrees. The cooling system starts operating for temperatures over 24 degrees and the heating system starts operating for temperatures below 20.

The design functions are given in Table 5.6. Basic elements such as doors, windows, and flat roofs with no skylight are used.

In this experiment the only change applied to Table 5.1 is generation number which is changed to 25. The reason is that GP converges quickly to minimum energy for cooling.

Figure 5.1 represents the average of the mean and average of the best fitness of the population. The average of the population fitness converges to zero amount of energy after six generations. The best model with no energy consumption for cooling is found in generation 1 in all trials.

Figure 5.2 illustrates one of the best models of 10 trials. All the 10 best individuals

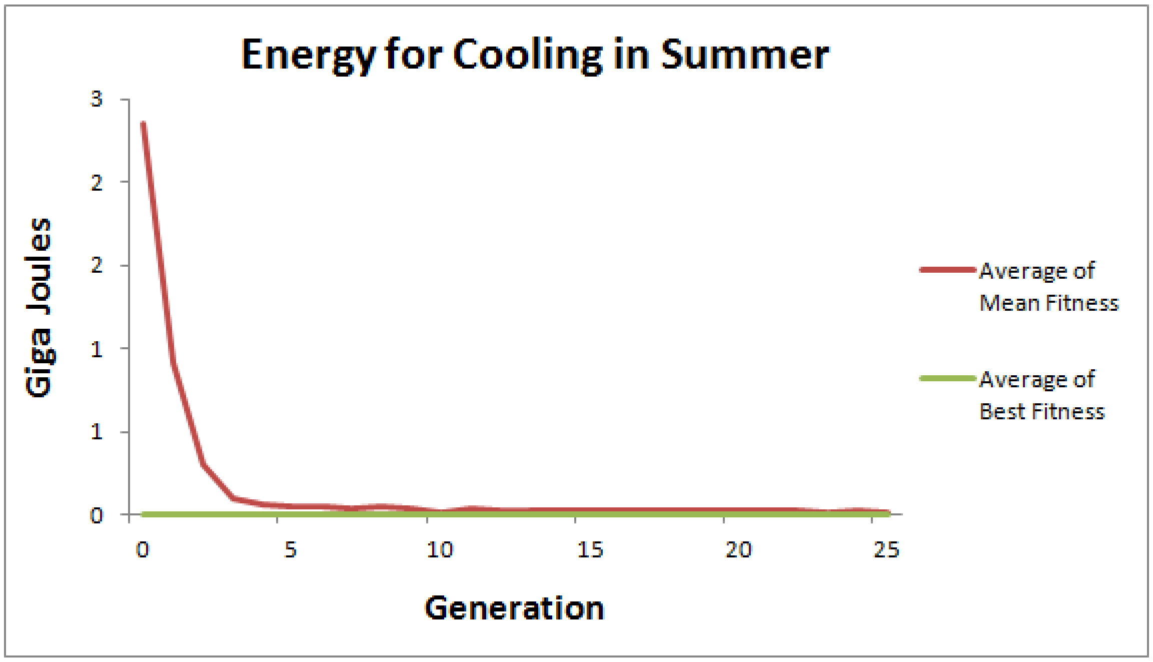


Figure 5.1: Population best and average for cooling energy in summer.

of the 10 trials have zero energy consumption for cooling. This shows that this problem is an easy one for GP. Also a material analysis on the best model of each trial shows that no specific material for walls, roofs, and windows are preferred. Some of them have single pane glasses for windows and some have double pane glasses with Argon or air in between. Some walls have insulation and some do not have. As we mentioned, this experiment was an easy experiment for GP and that is the reason why no specific material is preferred. In the next experiments, GP shows that it inclines to specific material and construction based on the problem and its objectives.

Figure 5.2 illustrates one of the best models of 10 trials. All the 10 best individuals of the 10 trials have zero energy consumption for cooling. This shows that this problem is an easy one for GP. A material analysis shows that no specific material is preferred for walls, windows, floors, and roofs. In other words, GP can design models with different combination of materials with no need of cooling energy for summer. The main reason is that thermal and ventilation performance are affected by many variables such as air distribution, type, size, and location of the heating and cooling system [30]. Also internal architecture and internal objects such as furniture and other commodities affect air distribution and condition. None of these parameters are considered in our designs and this adds inaccuracy in model evaluation. All the best models have similar styles of having no or very tiny windows facing south. In addition, most of the other walls have no windows to prevent heat transfer from outside.

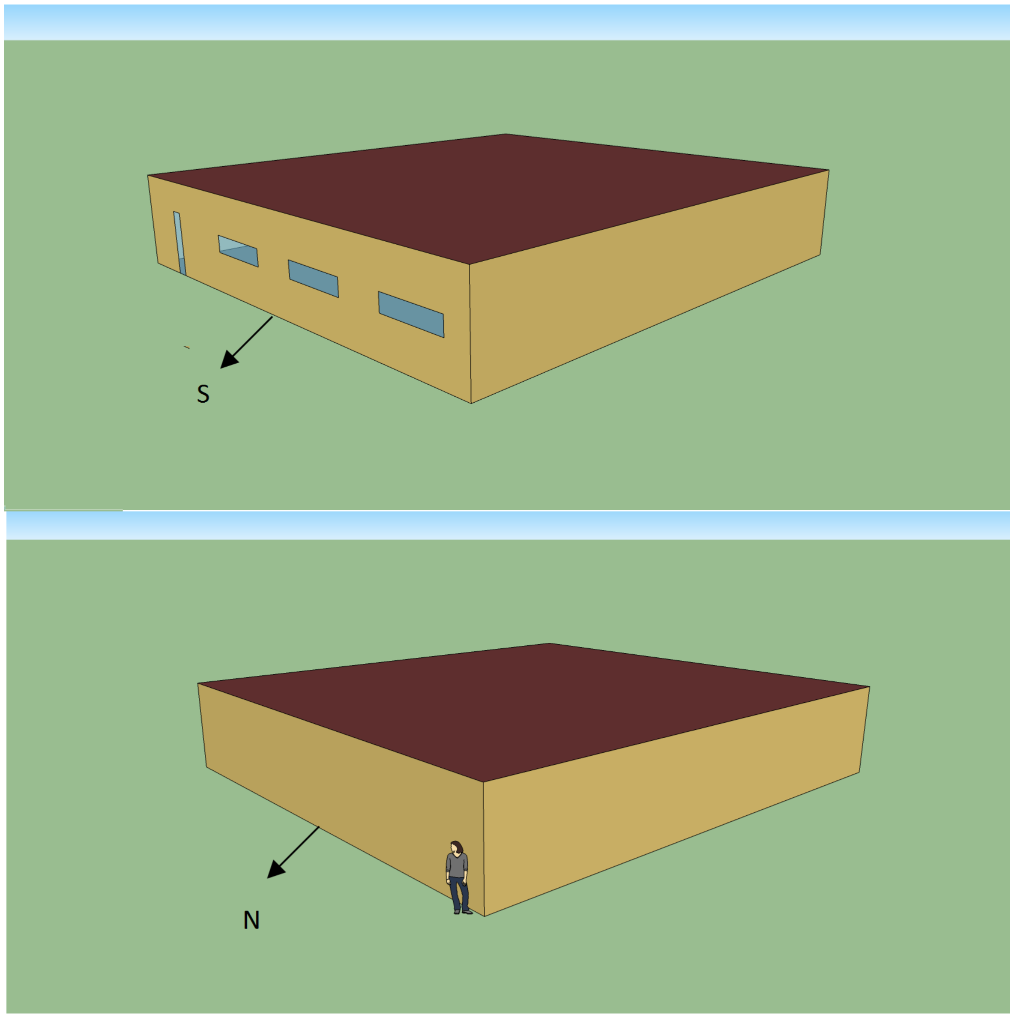


Figure 5.2: Single objective: the best model for cooling

5.1.2 Energy for Heating in Winter

In the last experiment, minimizing cooling energy during summer was the goal and 3D models with no need for cooling energy were obtained. Next, we aimed to find the best model that minimizes energy usage of heating in winter. Months January, February, and March are chosen as winter.

Figure 5.3 represents the average of the mean and average of the best fitness of the population. The best model of all the trials needs approximately 10 giga Joules energy to keep the building hot enough ($T_{min} = 20\,°\mathrm{C}$) during winter.

Figure 5.4 illustrates the best model of 10 trials. There are no windows in any side and there is a very tiny door at the front. This model shows that windows lose a lot of heat. Although windows let Sun heat the place, days are short during winter and nights are long. Therefore, window heat gain during days does not compensate for window heat loss during nights.

All of the best individuals from the last generation of all trials were trivially small 3x3x2 houses. This shows that smaller buildings have less need of energy for heating during winter. Wall_2 (wood, plywood, insulation, and gypsum) is selected by GP in all 10 best solutions. This wall is the wall with the smallest U-factor. U-factor describes how well a wall or in general a building element conducts heat. Materials with lower U-factor have lower heat transfer. The soft 4 millimeters wooden door (Door_1) is chosen in all the best individuals. This door also has the smallest U-factor among all doors in our system. The preferred roof by GP for this experiment is Roof_1 which has the biggest U-factor. Floor_2 which has the biggest U-factor in our system is also the preferred floor for the 10 best solutions.

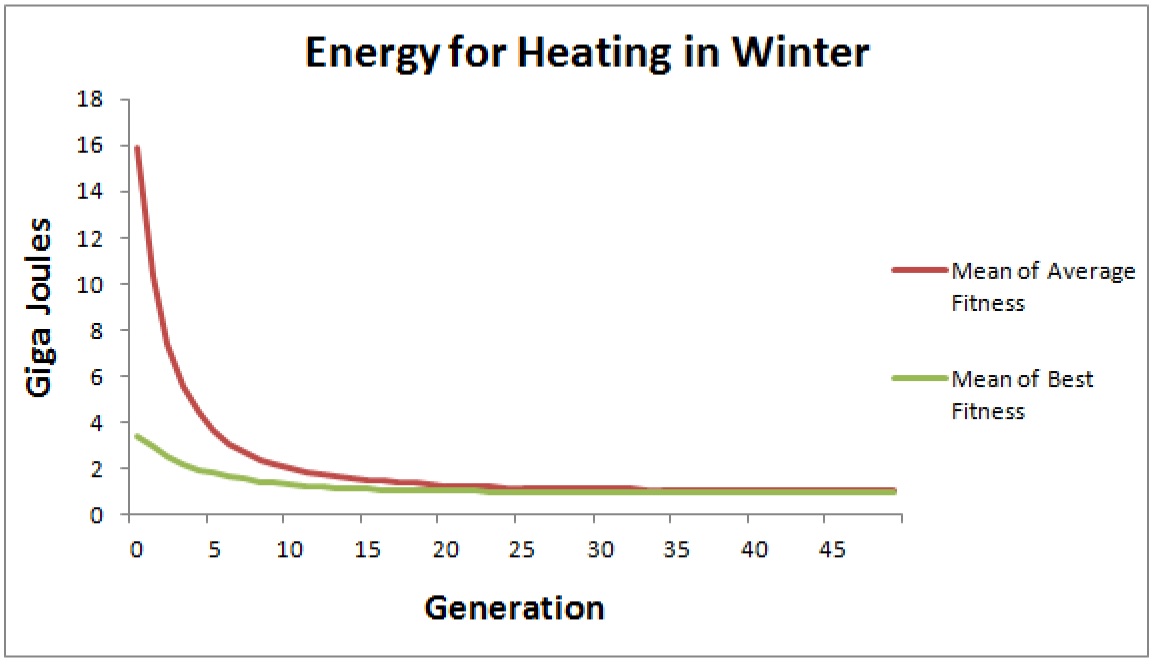


Figure 5.3: Single objective: best and average of the population for heating energy in winter over generations

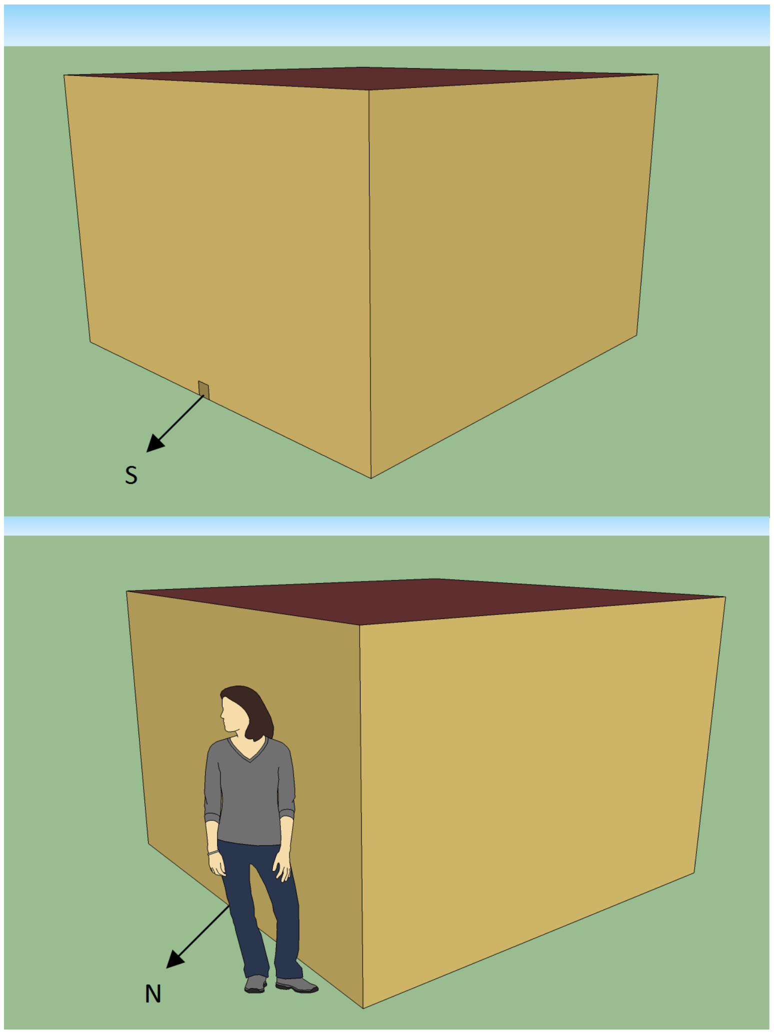


Figure 5.4: Single objective: the best model for heating in winter.

5.1.3 Window Heat Gain in Winter

In the last two experiments, the goal was to minimize energy usage for cooling and heating. Here, the goal is to maximize window heat gain in winter. The design functions of this experiment are given in Table 5.6.

In Figure 5.5, the average of the mean and average of the best fitness of the population is shown. The average of the population fitness converges to 21 KW heat gain. The best model gained 22240.5 wattage heats from sun light.

Figure 5.6, shows the window heat gain of the best models during a year. The solar heat gain is relatively high in summer and using a lot of energy to cool down the building during summer is needed.

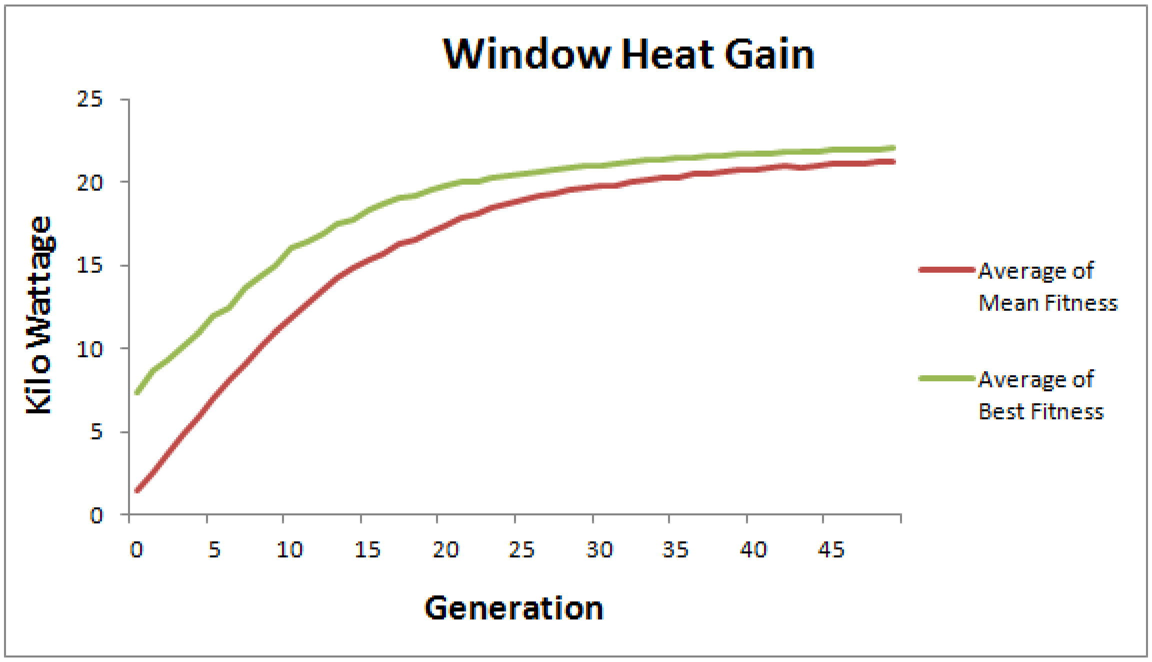


Figure 5.5: Single objective: best and average of the population for heat gain in winter over generations

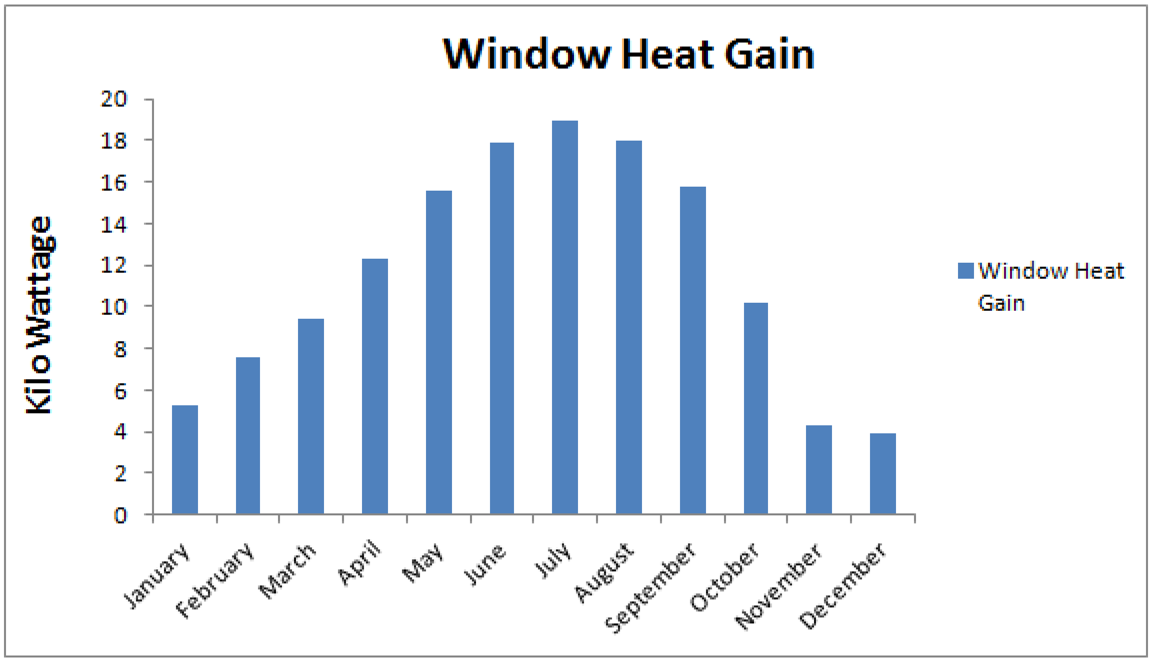


Figure 5.6: Window heat gain of the best model.

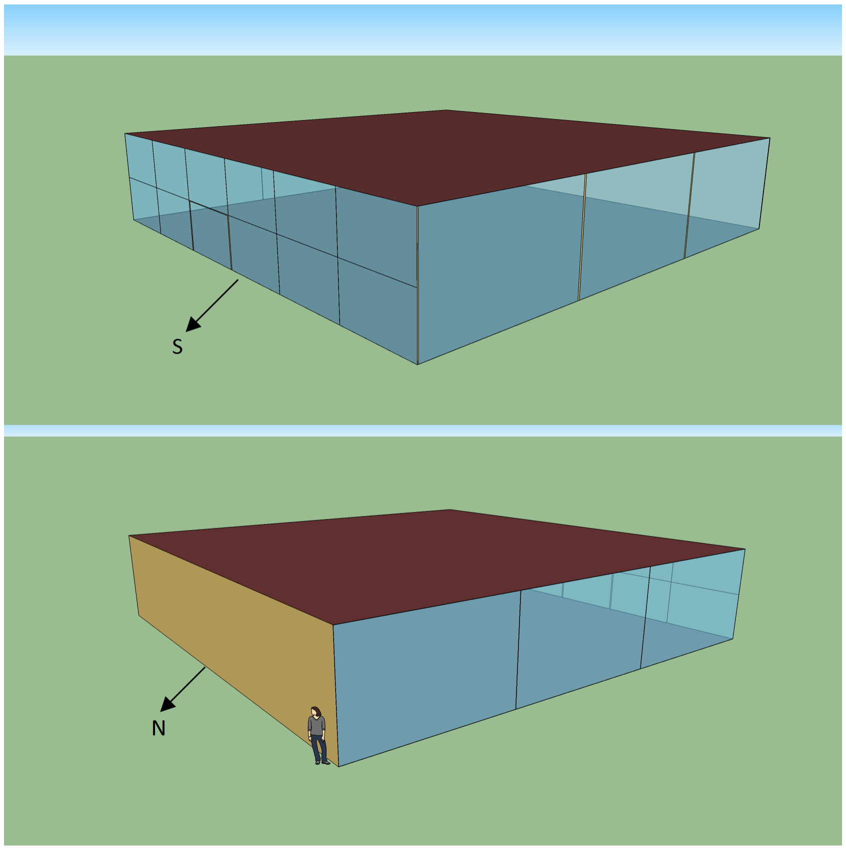


Figure 5.7: Single objective: the best model for heat gain in winter

Figure 5.7 illustrates the best model from 10 trials. Huge windows facing south, west, and east gains a lot of heat from sunlight. This model might be a good option for living in winter, but 24 GJ energy is needed to keep this building cool enough during summer.

A hand edited model identical to the best model with the difference of having an additional window that covers the wall facing to the north was tested. Gaining 21301.91 wattage heat in winter proves that adding windows facing to the north does not help a lot in gaining more heat. In fact, in calculating heat gain, short-wave radiation from the building transmitted back out the window is deducted from the actual heat gain and that is the reason why having windows facing to the north causes less heat gain during winter. Figure 5.8 presents a comparison of heat gain of the best individual during generations and its hand edited version.

Some consistency in all the solutions were observed. For example, each run's best model used its maximum size for the building: 20x20x4 meters.

All of the best GP trees of 10 trials were examined to find out which materials are useful for windows, roofs, floors, and doors. Window_2 (double pane window with

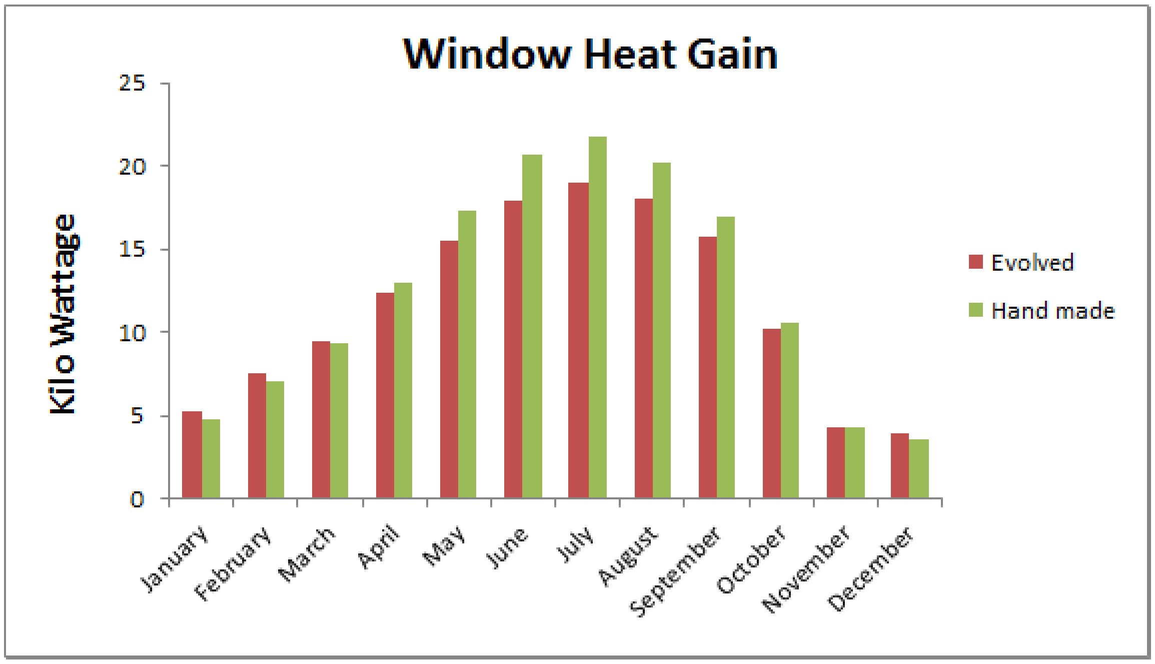


Figure 5.8: The comparison between two models: the best of the all 10 trials and a hand edited model which is identical to the best, except that it has windows facing north. Window heat gain during winter is less in hand edited model.

argon in between) is the preferred window. These types of windows have the second smallest U-factor among all other windows in our system. This means that they are not good in conducting heat. Lower heat conduction keeps heat inside the building. Although these windows have second best U-factor, they have better solar heat gain coefficient in all double pane windows. All of them used Wall_4 (two layers of gypsum with air gap in between) for the wall facing north which has the biggest U-factor of all walls in our system. All sides of the building except the north one are filled with windows. All the doors used in the best models were glass doors which help gaining heat as well. All the floors were Floor_1 (concrete, and hardwood) which has the lowest U-factor in our system. Most of the roofs were made of the roof with the biggest U-factor.

5.2 Multi-Objective Experiments

In this section, more than one objective is considered. In each experiment, the objectives that are used will be identified. Our multi-objective technique is NRS which was discussed in Section 2.3.3.

5.2.1 Window Heat Gain in Winter and Annual Heating and Cooling Energy

Here, both annual heating and cooling energy, and window heat gain in winter is considered. Window heat gain and annual energy consumption are conflicting objectives, since maximizing window heat gain requires to have larger windows while energy consumption objective forces models to have smaller windows. The design functions are given in Table 5.6.

Figure 5.9 and 5.10 represent the average of the mean and average of the best fitness of the population in minimizing energy usage and maximizing window heat gain respectively. The best GP tree among the best trees of the last generation of 10 trials is selected by NRS. The best model with 16 giga Joules annual energy consumption is illustrated in Figure 5.11. The best GP tree has 7349 wattage window heat gain in winter. Its monthly window heat gain is presented in Table 5.12. The best model has windows facing south and no windows on the other sides. Having windows facing south guarantees heat gain needed for winter and having no window on the other sides help in having less need for cooling and heating energy.

The size of the all best models is approximately 20x20x2 meters which means that the maximum possible width, and length and minimum possible height for sizes are chosen by GP. Maximum length causes more windows in south which helps in gaining solar heat. Maximum width helps in capturing solar heat in a bigger area, which mean less heating energy. Having minimum height also helps in need for less heating and cooling energy

A material investigation of the best models shows that materials are a mixture of materials chosen in previous experiments. All the walls except the south one are Wall_2 (wood, plywood, insulation, and gypsum). This wall is the same as the walls in energy for heating during winter experiment. These walls have the smallest U-factor and are not good in heat conduction. Therefore, they keep heat inside. The wall facing south in most cases is the same as the wall we mentioned here. Some of them are the wall with the second U-factor. As most of the wall facing south is windows, U-factor of the best wall and the second best wall does not have a big role in energy

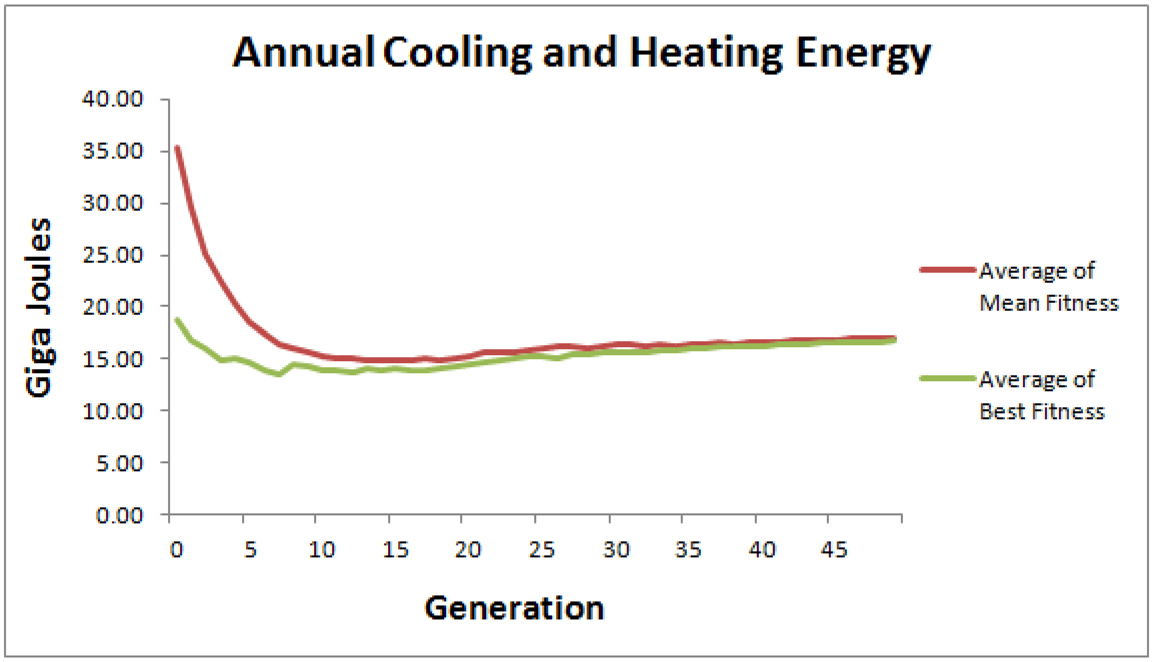


Figure 5.9: Heat gain and annual energy consumption experiment: best and average of the population for annual energy over generations

efficiency in this case. Window_2 (double pane window with argon in between) is the selected window by the best 10 solutions. These windows are identical to the

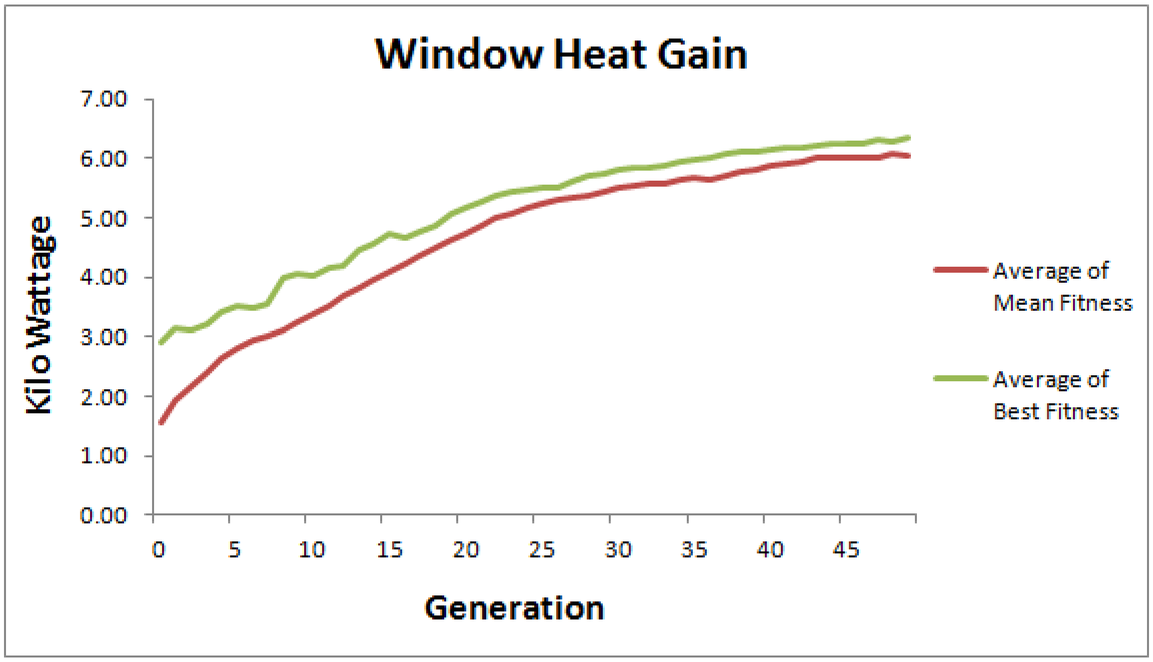


Figure 5.10: Heat gain and annual energy consumption experiment: best and average of the population for heat gain in winter over generations

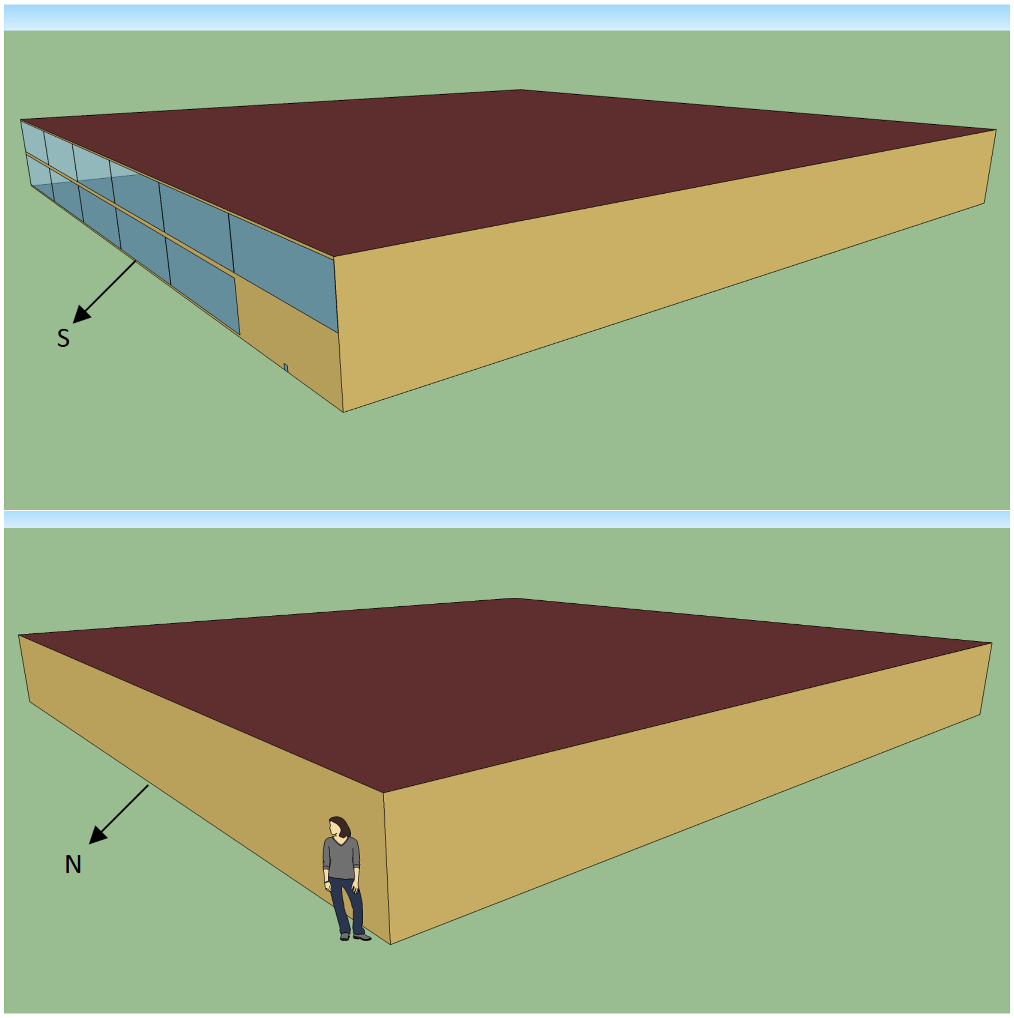


Figure 5.11: Heat gain and annual energy consumption experiment: the best model

windows used in maximizing window heat gain during winter experiment in Section 5.1.3. The door also has the lowest U-factor among other doors in our system. The roof and the floor are the ones with biggest U-factor (Roof_1, and Floor_2).

Figure 5.12 presents monthly window heat gain of the best model of all runs. Non-trivially, the values in Figure 5.12 are lower than the values in Figure 5.6. Figure 5.13 shows the annual energy consumption of the best model of the 10 trials. This model is highly energy efficient in summer since no energy is consumed from June to August.

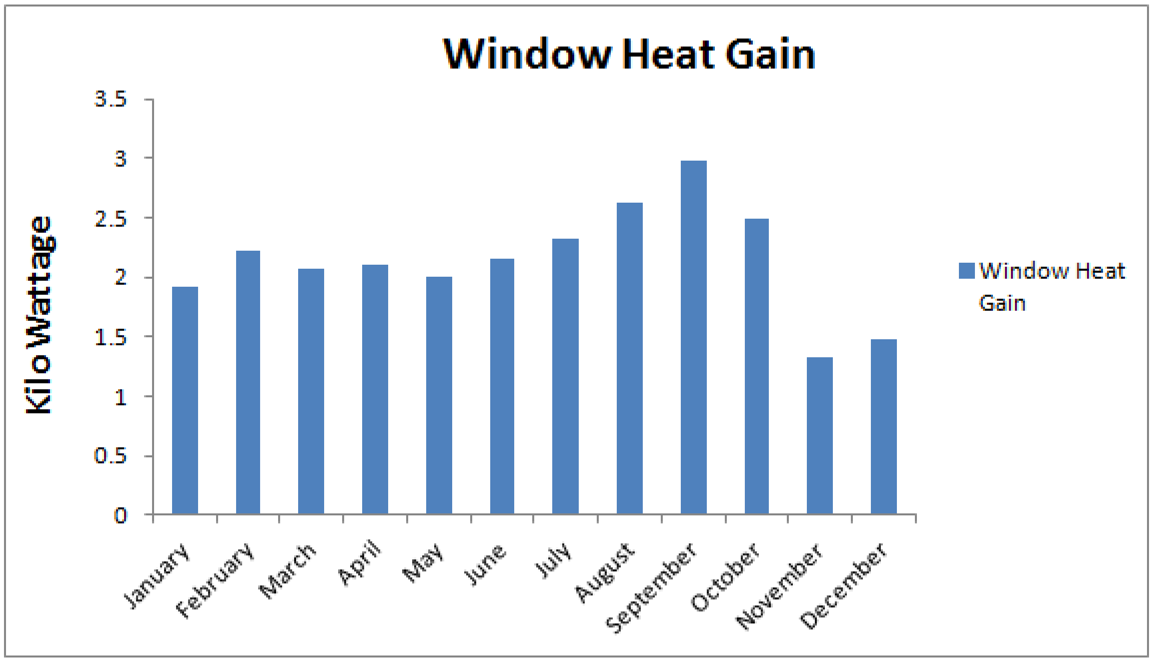


Figure 5.12: Best model heat gain.

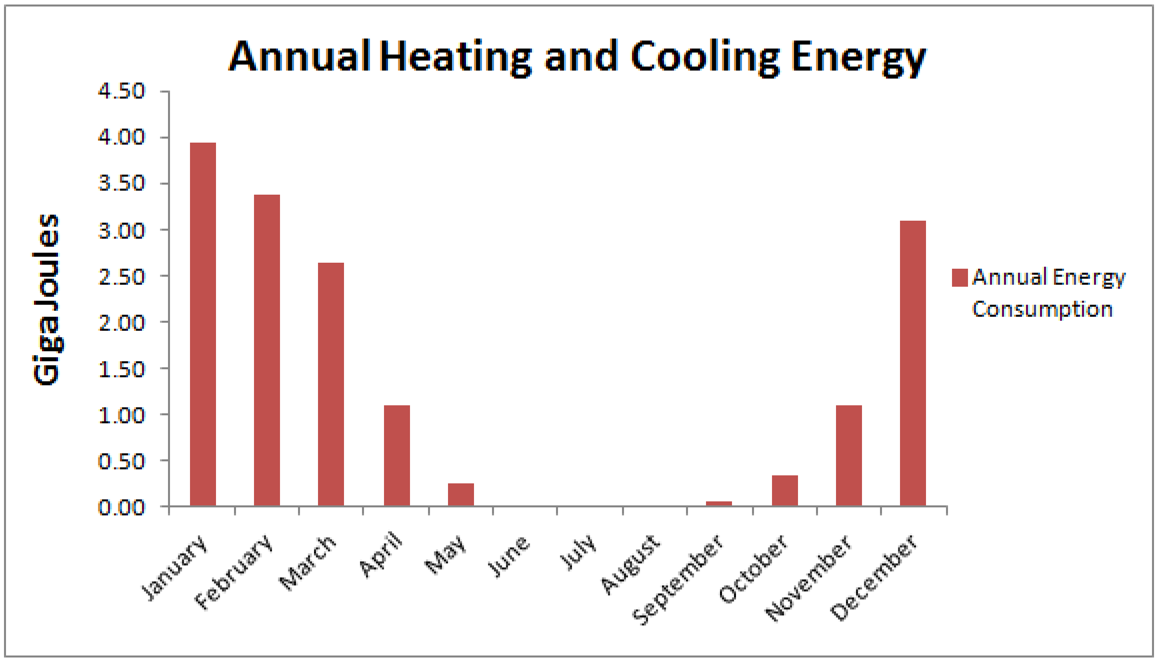


Figure 5.13: Best model annual energy consumption.

5.2.2 Window Heat Gain in Winter, Annual Heating and Cooling Energy, and at Least 25% Window Area

In previous experiments, energy based objectives were used. Next, we add a non-energy based objective, which is that each wall has to have at least 25% window area. If a wall has less than 25% window/wall ratio, the difference is called an error. The aggregation of errors' squared is another objective used in this experiment. We call this objective sum of errors (SOE).

$$SOE = \sum_i (0.25 - window/wall)^2$$

where i $\in$ { walls that have less than 25% windows }.

The design functions of the experiment are given in Table 5.6 and are the same as previous experiments.

Figure 5.14 shows monthly heat gain of the best model of the 10 trials. Comparing Figure 5.14 and Figure 5.12, shows that having more windows in each side of the building causes having more heat gain during each month. By having big windows in the south of the building, window heat gain in winter is high. Figure 5.15 shows the annual energy consumption of the best model. The best model gains a lot of heat from windows during summer and 0.38 giga Joules energy is needed to cool down the place during summer. The energy used for the best model of the last experiment is 0.07 giga Joules which is far less than 0.38 giga Joules of this experiment.

Figure 5.16, 5.17, and 5.18 depicts annual energy consumption for cooling and heating, window heat gain during winter, and SOE over generations.

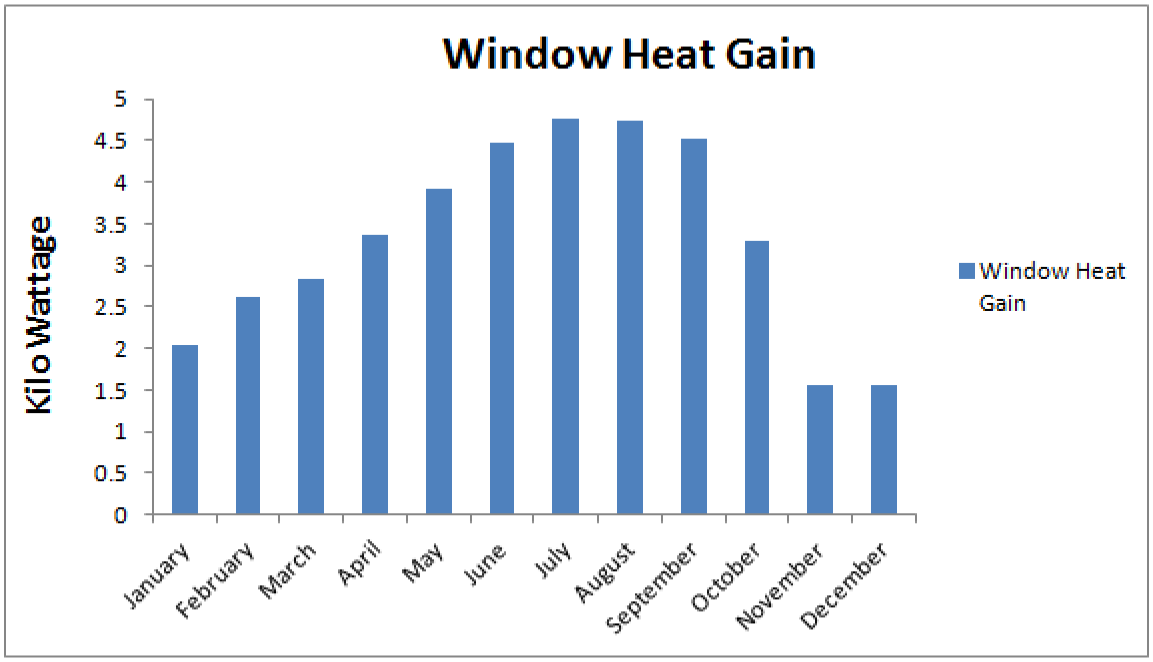


Figure 5.14: Window heat gain of the best model

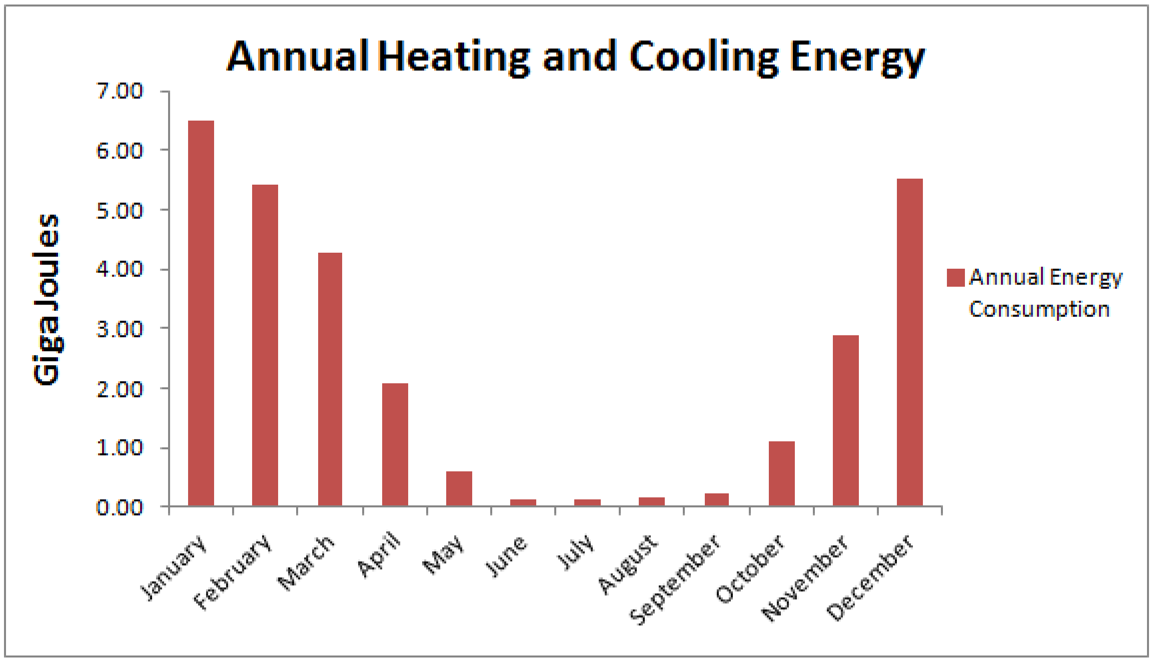


Figure 5.15: Annual energy consumption of the best model

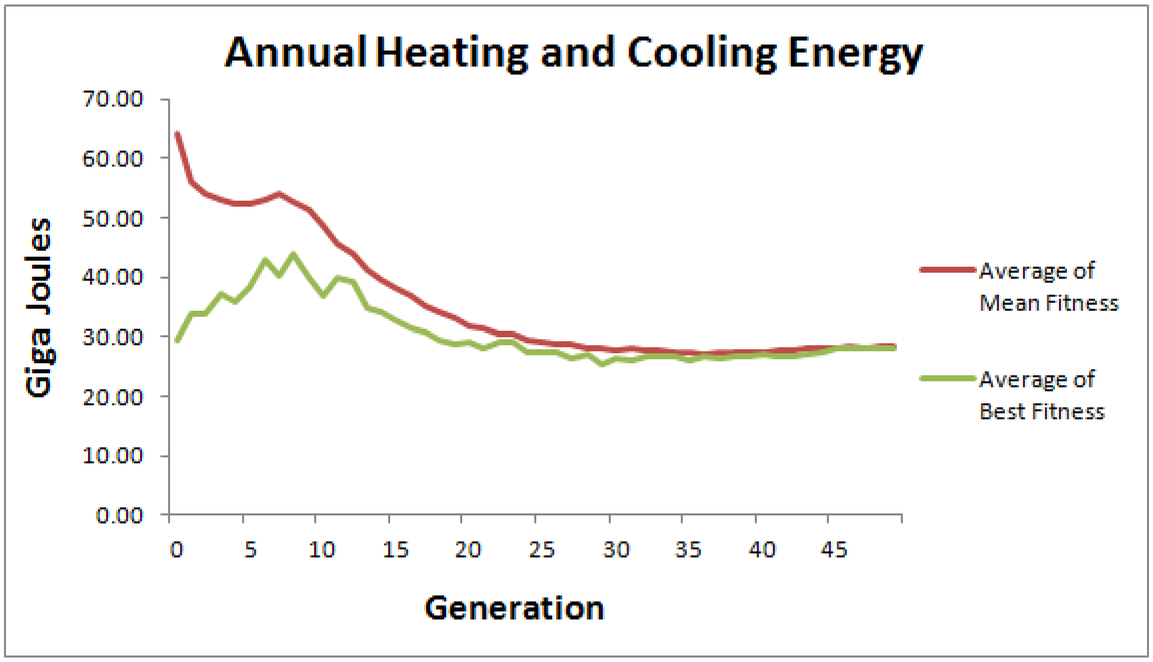


Figure 5.16: Window heat gain, annual energy, and at least 25% window on each side experiment: annual energy consumption over generations is illustrated.

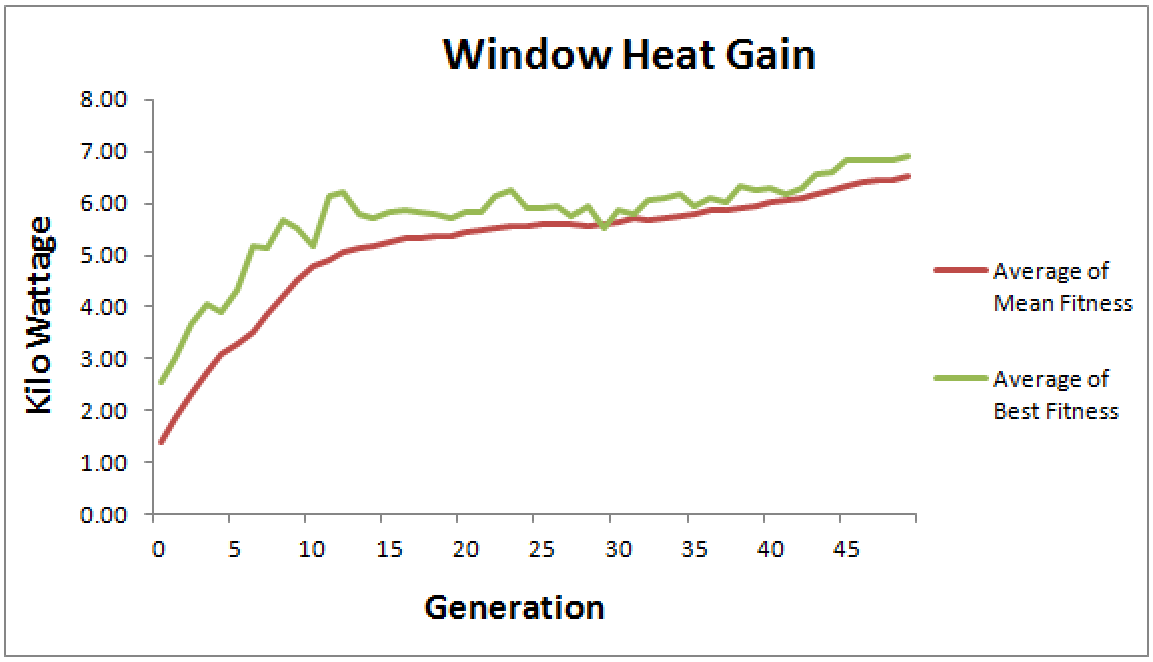


Figure 5.17: Window heat gain, annual energy, and at least 25% window on each side experiment: window heat gain over generations is illustrated.

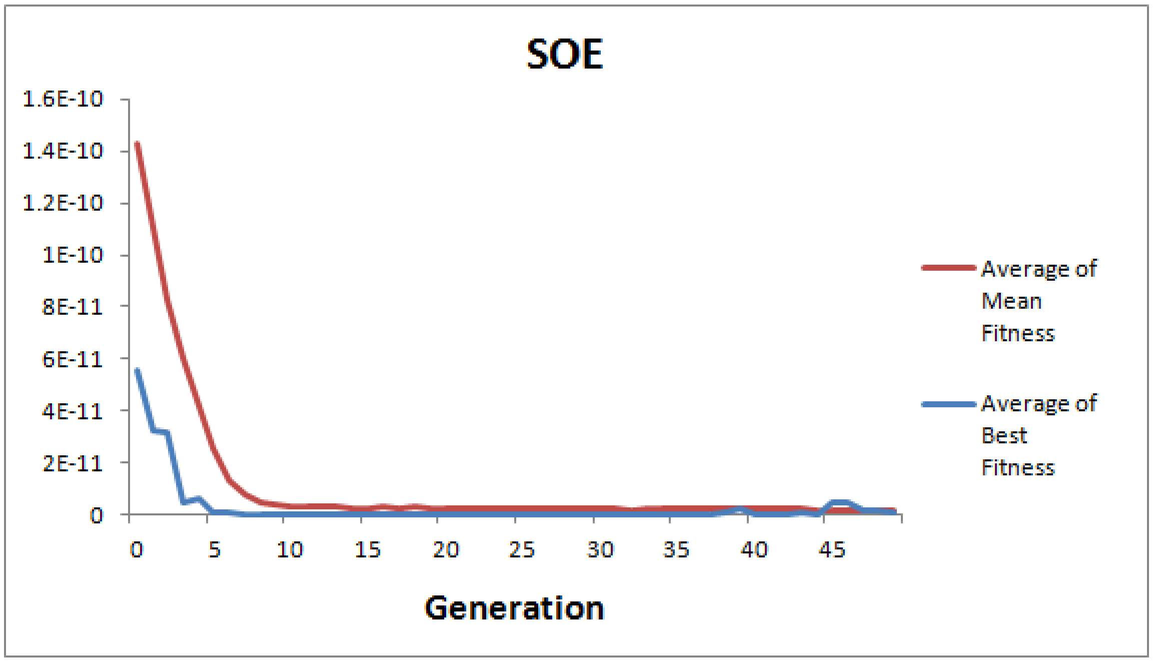


Figure 5.18: Window heat gain, annual energy, and at least 25% window area on each side experiment: SOE is shown over generations.

The best model of all trials is shown in Figure 5.19. The size of this building is 20x20x2 meters which has the maximum width, and length and minimum height. The best in case of energy usage without considering window heat gain and at least 25% window area is shown in 5.20 and its size is 12x7.5x2. This is reasonable that this model is the best in case of energy, because it was the smallest one compared to other best individuals of the last generation of all runs.

After investigating materials, we found out the best window in case of U-factor and solar heat gain coefficient(Window_2) is selected for all best individuals. But the best wall in case of U-factor is not selected. The reason is that all the best 10 solutions in this experiment are having at least 25% windows on each side. This means that heat gain is increased. Therefore, there is more need to cool the building down during summer. On the other hand, buildings do not want to lose heat during winter. Therefore, the wall with the biggest U-factor is not selected as well. The selected wall is Wall_5 (dense brick, insulation, concrete, and gypsum) and it is the third best in case of U-factor. Floor_2 (concrete, and hardwood) with the biggest U-factor is selected by evolved solutions.

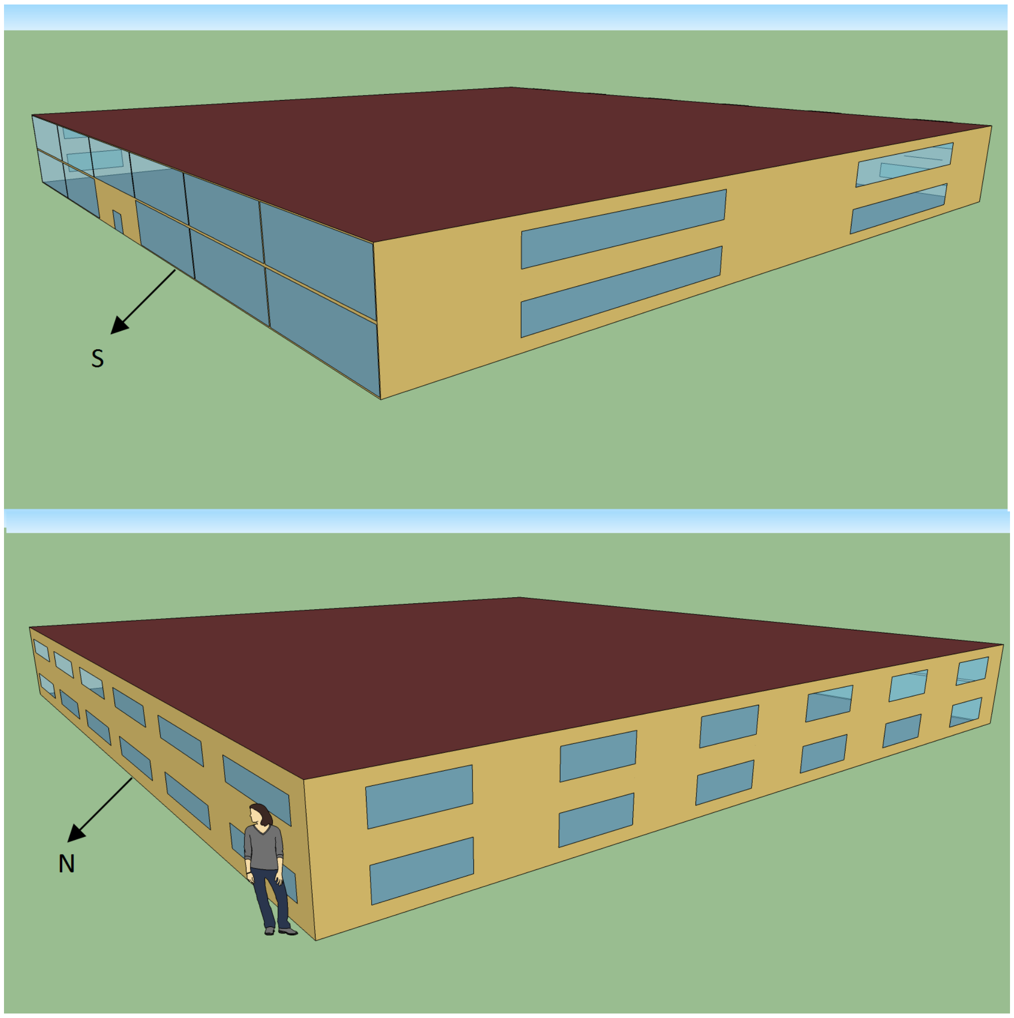


Figure 5.19: The best model of the 10 trials. In addition to window heat gain in winter and annual energy consumption, at least 25% window area on each side is accomplished. All sides except south have 25-26% windows. Size of this model is 20x20x2 meters.

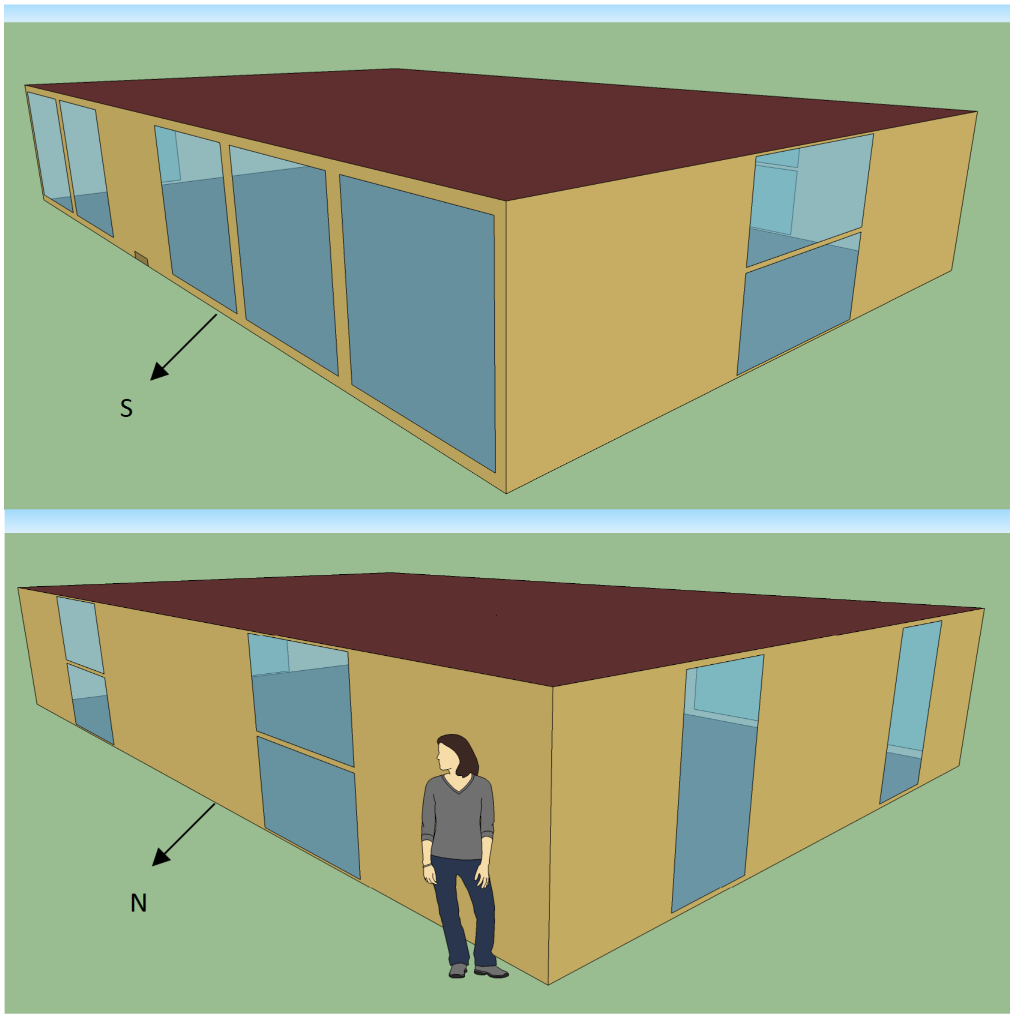


Figure 5.20: The best model of the 10 trials in less energy consumption. In addition to window heat gain in winter and annual energy consumption, at least 25% window on each side is needed. Size of this model is 12x7.5x2.

5.3 Summary

This chapter examined the impact of different energy based objectives on models. The last experiment also added a window constraint to add more windows.

The first experiment showed that finding models with minimum cooling energy usage is an easy task for GP. The second experiment determines that if the goal is to minimize heating energy in winter, then a very tiny house with no window is the solution. To maximize window heat gain in winter, having windows in all sides except north is the result. Since in window heat gain calculation, short-wave radiation transmitted back from windows is subtracted from other parameters, window heat gain of a specific window can be negative. That is the reason that no windows appeared at the north wall for window heat gain experiment. The experiment with a combination of annual energy and window heat gain determined that models with windows facing south and no windows on other sides are the best models. The last experiment showed the impact of energy based constraints with window constraint on models. Models still inclined to have more windows on the south wall and about 25% on other walls. All material of the last two experiments remained the same except walls.

Chapter 6

Experiments for Different Locations

This chapter overviews the impact of different geographic locations on models and evolution. Different geographic locations used here are as follows:

- Toronto: baseline location.
- Anchorage, Alaska: north cold.
- Eldoret, Kenya: near the equator.
- Las Vegas, Nevada: hot location.
- Melbourne, Australia: south hemisphere.

Weather data files are taken from [36]. Average temperature of the locations used in this chapter is given in Table 6.1.

Location	**January**	**July**
Toronto	-6	21
Eldoret	17	17
Anchorage	-2	14
Las Vegas	3	20
Melbourne	18	8.5

Table 6.1: Average 24 hour temperature of January and July for 5 cities in Celsius.

6.1 Experimental Setup

All experiments in this chapter uses GP parameters mentioned in Table 5.1. The only difference is that the generation number is increased to 100.

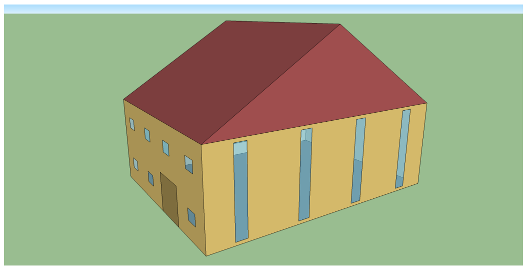
Having looked at results from previous chapter, we realized that impractically small models were generated. Design parameters are changed for this section to have more reasonable models in terms of size and shape. Table 6.2 presents new parameters used for all experiments in this chapter. Minimum floor width and length is increased to 10 meters. The minimum and maximum floor height is increased to 4, and 8 meters respectively. Door and roof constraints are added to the system as well. The minimum and maximum door's height are 2 and 8 meters. The minimum and maximum door's width are 1 and 6 meters. If roofs are not flat, their heights are limited between 3 and 10 meters.

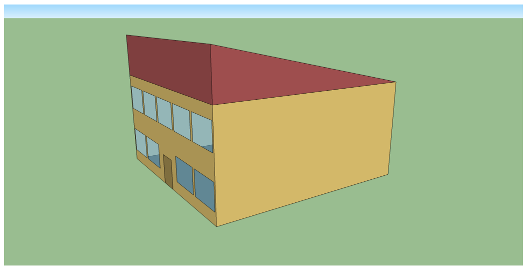
Parameter	**Value**
Maximum Floor Length	20 meters
Minimum Floor Length	10 meters
Maximum Floor Width	20 meters
Minimum Floor Width	10 meters
Maximum Floor Height	8 meters
Minimum Floor Height	4 meters
Maximum Door Height	8 meters
Minimum Door Height	2 meters
Maximum Door Width	6 meters
Minimum Door Width	1 meters
Maximum Roof Height	10 meters
Minimum Roof Height	3 meters
Maximum Number of Rows on a Facade	2
Maximum Number of Columns on a Facade	6

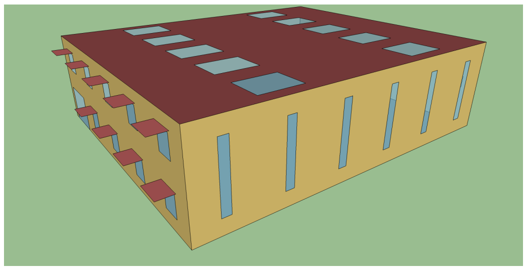
Table 6.2: Design Parameters

GP Functions	Add Root, Add Cube, First Floor, Add Door Grid, Add Grid, Add Door, Add Window, Add Window Overhang, Add Empty Grid, Add Simple Roof, Add Skylight, Add Gabled Roof, Add Gabled Roof2, GP mathematical functions
GP Terminals	ERC, Int_ERC

Table 6.3: Design functions: these functions and terminals are used in all experiments of this section.


(a) Gabled roof type 1.


(b) Gabled roof type 2.


(c) Overhangs and skylights.

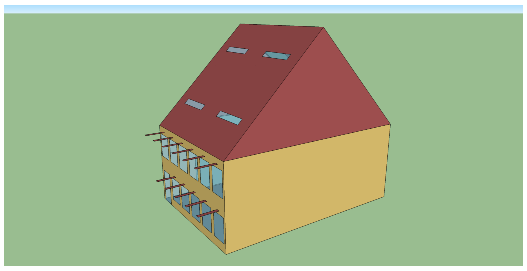

(d) A gabled roof with skylights.

Figure 6.1: New design elements used in this chapter

Table 6.3 presents different design functions and terminals can be used by the system. Overhangs, skylights, and two more kinds of roofs are added to Table 5.6. Figure 6.1 illustrates new design elements.

Objectives of all experiments are as follows:

1. Window heat gain in winter.
2. Annual energy consumption.
3. At least 25% window area.

The multi-objective strategy is NRS. For each location, 10 trials have been performed. Materials used here are identical to the materials used in Chapter 5 and they are listed in Table 5.3 and 5.4.

6.2 Results

6.2.1 Population Performance Analysis

Table 6.4 shows the window heat gain and annual energy of the population average, best solutions average, and the best solution for each location. Anchorage and Eldoret are respectively the worst and the best cities in window heat gain in winter. The best model in window heat gain for winter among all 5 best models belongs to Las Vegas

	Population		Avg Top Solutions		The Best Solution	
Location	Heat Gain (W)	Annual Energy (GJ)	Heat Gain (W)	Annual Energy (GJ)	Heat Gain (W)	Annual Energy (GJ)
Toronto	20056	120	20306	117	15768	78
Las Vegas	44299	112	45389	112	65433	199
Eldoret	49540	58	50293	58	46453	41
Anchorage	7426	97	7537	96	2339	43
Melbourne	24726	51	25005	51	16408	20

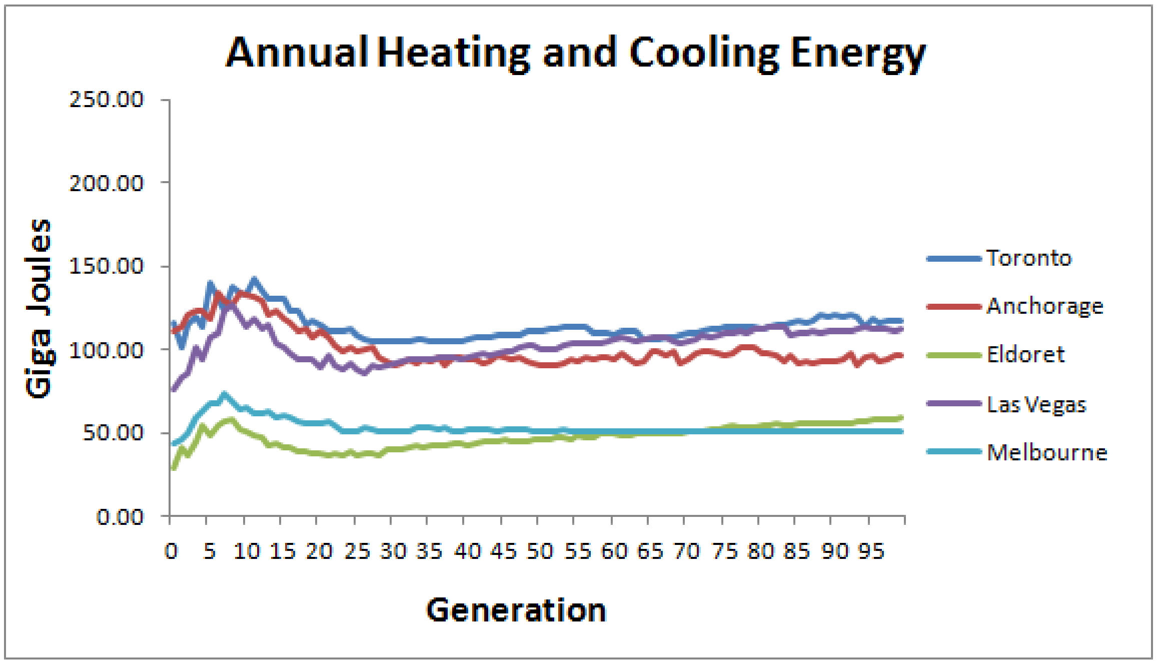
Table 6.4: Population heat gain, Population annual energy consumption, best solutions' heat gain, best solutions' annual energy, best solution heat gain, and best solution annual energy. (Population average is for final generation)

with 65.5 KW heat gain and the worst belongs to Anchorage with 2339 KW heat gain. Melbourne has the least energy consumption. Although the best model of Las Vegas has the most energy consumption among the best solution of all cities, Toronto models have the most energy consumption on average. An analysis of variance (ANOVA) of window heat gain for 5 cities shows that window heat gain of cities are significantly different. The same analysis have been done for annual energy and the result of that also shows that annual energy consumption of the cities are significantly different.

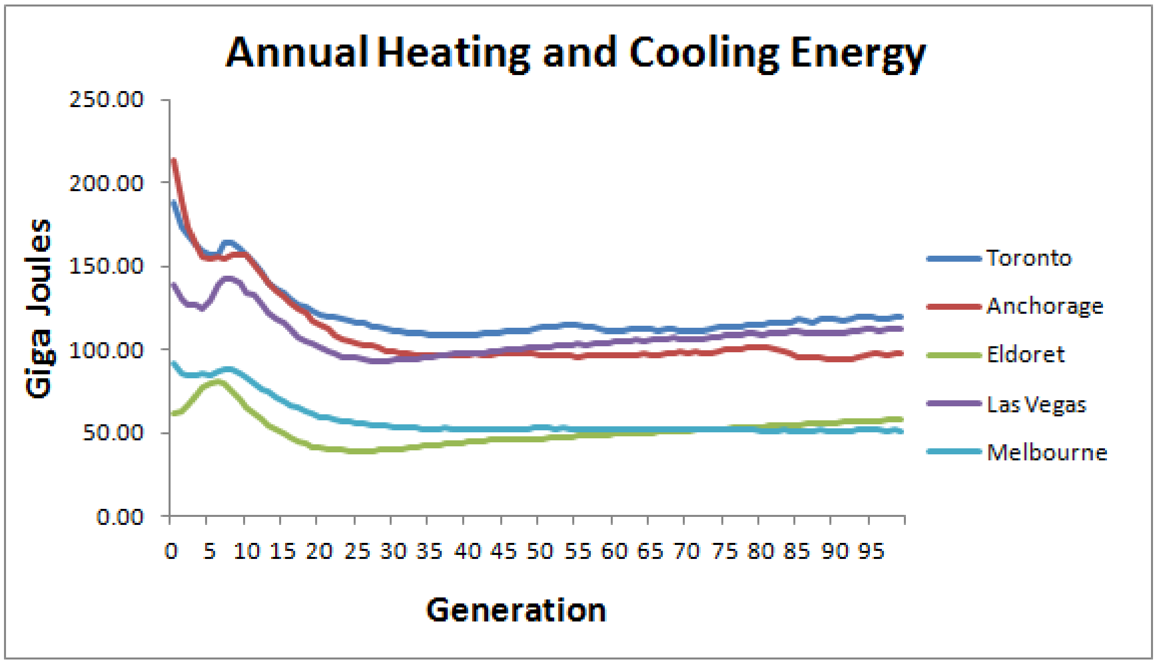
Figure 6.2 shows the population average and the best 10 solutions average of each city. All locations find their best model in aspect of least energy consumption in generation 23- 27 and then they become worse over generations. This shows the impact of other objectives on this objective. Window heat gain affects models to have larger windows and this is the reason that energy consumption becomes worst after generation 27. Figure 6.2 also approves that Toronto models have the most energy consumption and Melbourne models have the least energy consumption.

Figure 6.3 depicts the population average and 10 best solutions' average of each location in window heat gain during winter. January, February, and March are considered as winter for all models except Melbourne. For Melbourne, July, August, and September are considered as winter. For all cities, window heat gain constantly increases. Eldoret has the most window heat gain and Anchorage has the least window heat gain.

Figure 6.4 represents SOE objective. On average, models do not have difficulties in finding models which have low SOE. However, Figure 6.4a shows that the best 10 models for each city do not have necessarily SOE equal to zero.

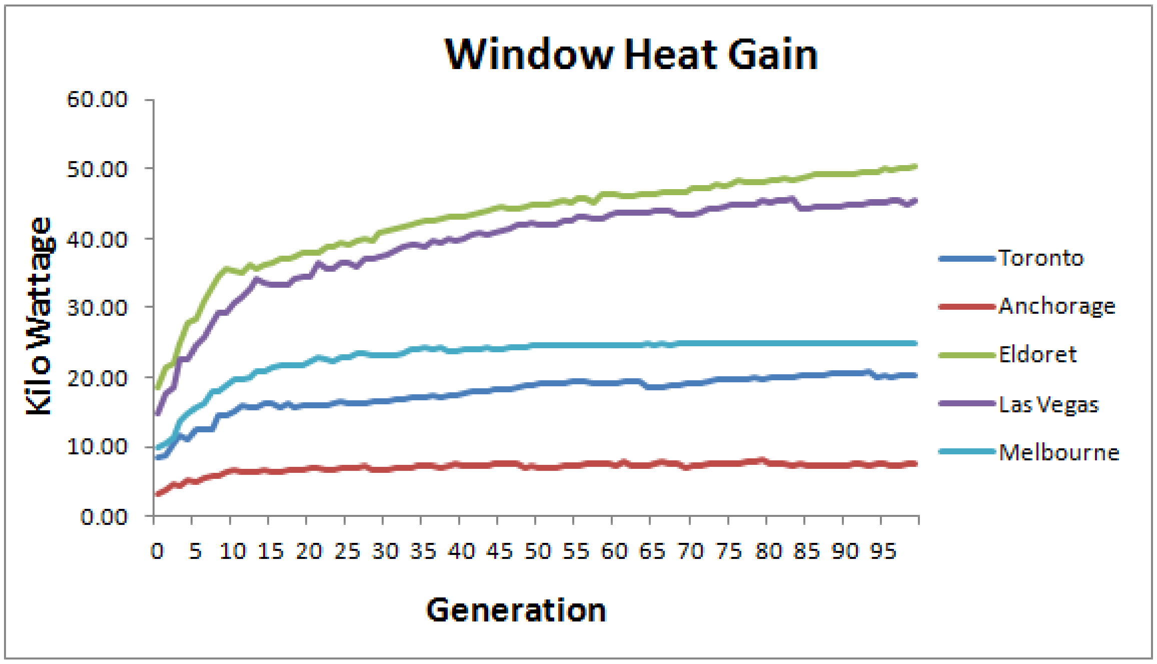


(a) Average of the best 10 solutions.

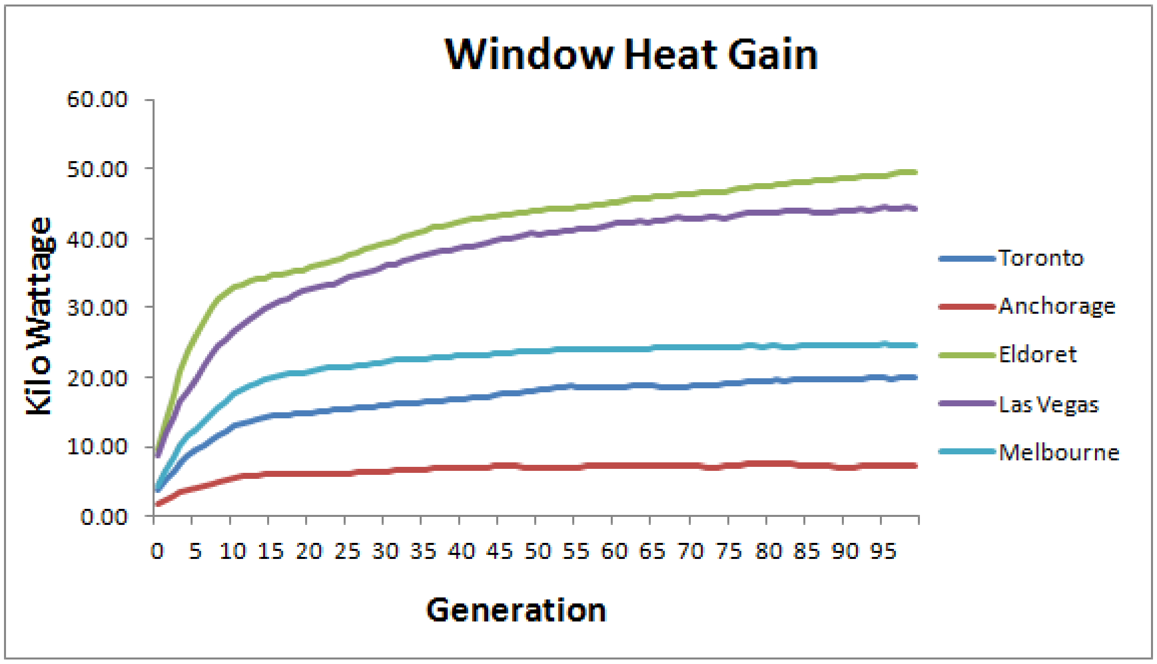


(b) Average of the population.

Figure 6.2: Five locations experiment: annual energy

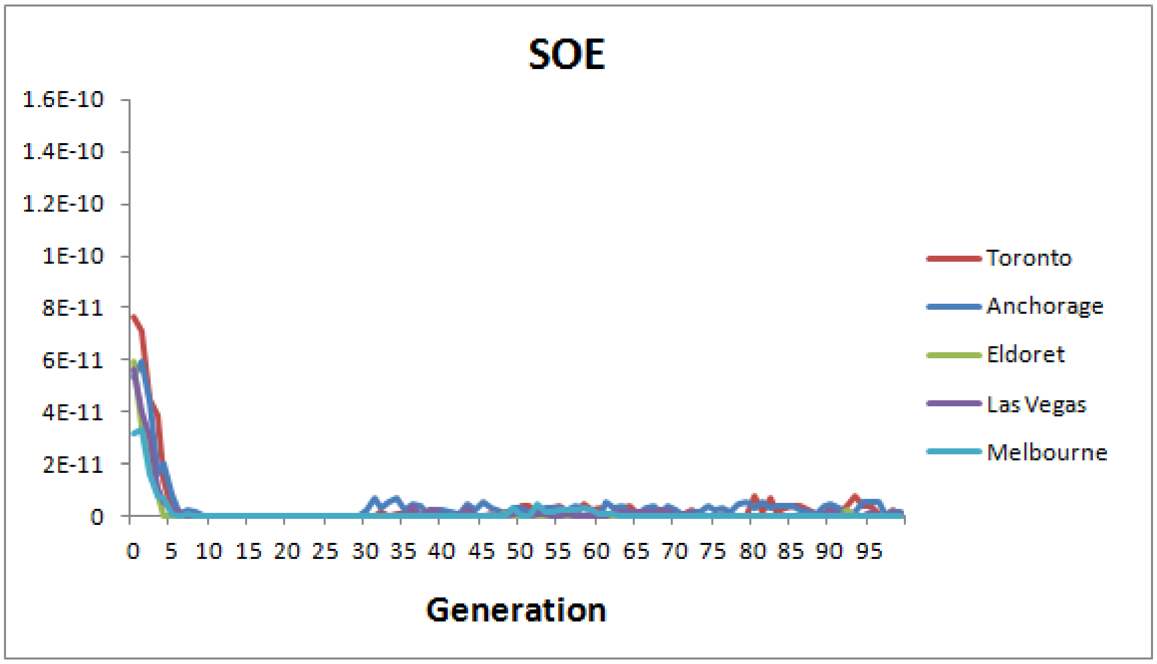


(a) Average of the best 10 solutions.

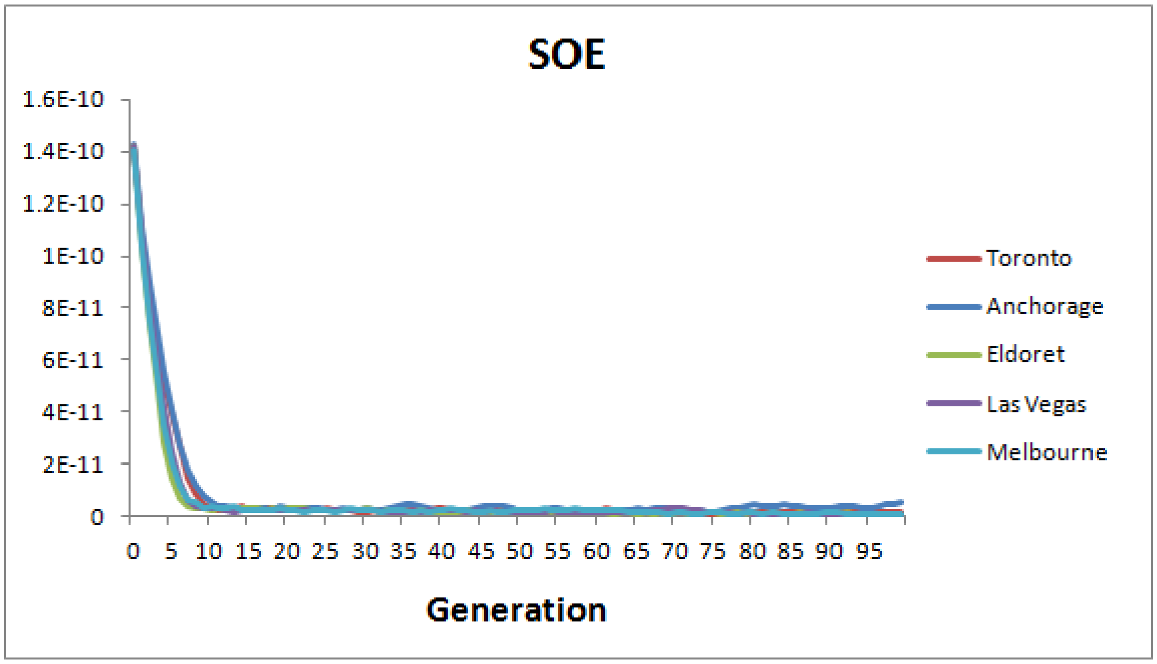


(b) Average of the population.

Figure 6.3: Five locations experiment: window heat gain



(a) Average of the best 10 solutions.



(b) Average of the population.

Figure 6.4: Five locations experiment: sum of errors

6.2.2 Best Models Analysis

Figure 6.5 shows an energy vs. heat gain scattered plot of the 50 best models of all 50 trials. Window heat gain has a direct relation to annual energy. More window heat gain causes more annual energy consumption. Therefore, a model with no window is the most energy efficient model. Most models in Figure 6.5 are undominated according to Pareto ranking. A Pareto ranking applied on the best 10 solutions of each city. Table 6.5 shows the number of undominated models out of 10.

According to Figure 6.5, Eldoret which is on equator has the most window heat gain on average. In this aspect, Anchorage is opposite of Eldoret. It has the least window heat gain on average with little solar heat gain in winter and more in summer. The model with the most energy usage belongs to Toronto (a). Also, the Melbourne model has the least energy usage (b).

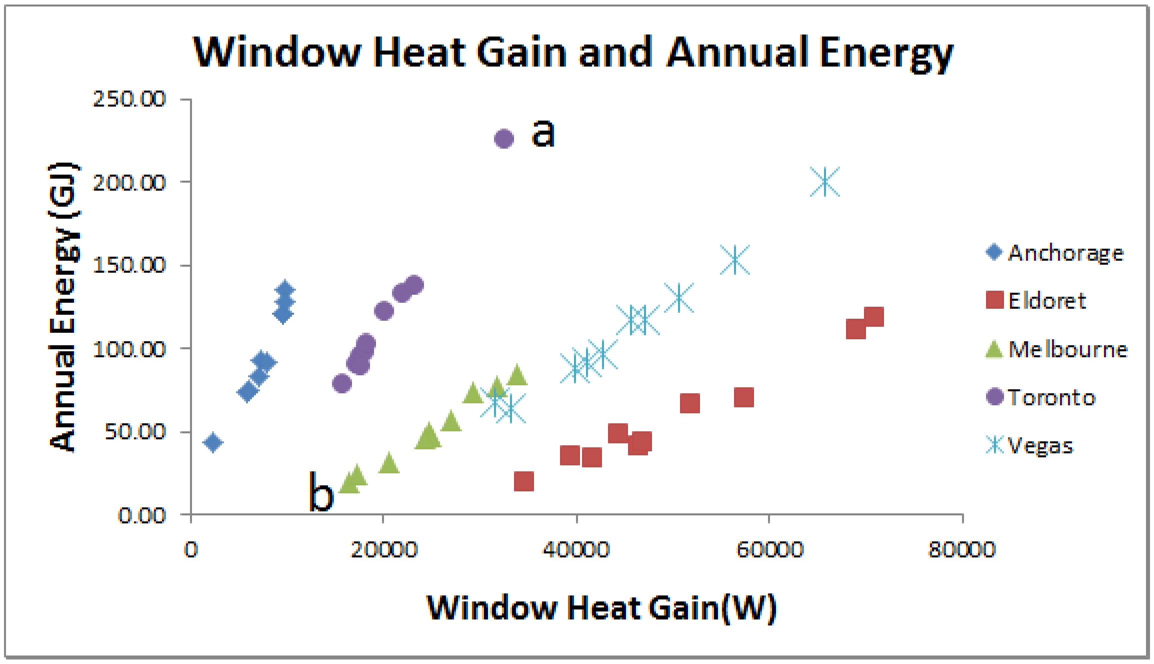


Figure 6.5: Window Heat Gain VS. Annual Energy

Location	Undominated
Toronto	9
Las Vegas	9
Eldoret	8
Anchorage	8
Melbourne	9

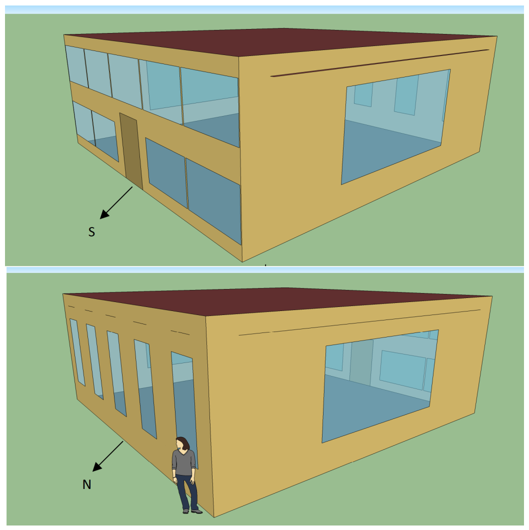
Table 6.5: Number of Pareto undominated models of the best solutions.

Figure 6.6 illustrates the 5 cities' best models. The best models for each follow specific styles. As an example, Figure 6.7 illustrates all 10 best model of Toronto. The pattern in Figure 6.7 is having the most windows facing south and almost 25% windows on other sides. Table 6.6 shows window area percentage of the models. Anchorage, Toronto, and Las Vegas are all on the northern hemisphere. They have the most windows facing south. Eldoret is located on the equator and have almost 50% windows facing west, east, and south and 25% window area on north. Melbourne is on south hemisphere and its models have the most windows facing north unlike the north hemisphere cities that have the most window area facing south.

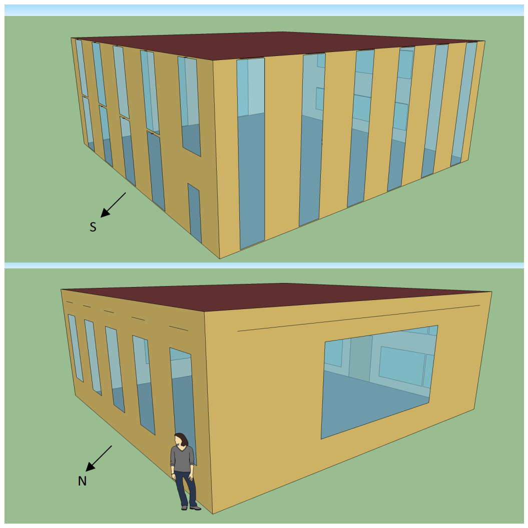
No skylight or non flat roofs are selected for the best 50 models. Some models have overhangs, but selected overhangs are tiny and do not have a great impact on window heat gain and/or energy efficiency.

Location	**South**	**West**	**North**	**East**
Toronto	94	27.5	24	35
Las Vegas	87	28	25	28
Eldoret	45	52.5	27.5	55
Anchorage	89	26	22.5	28
Melbourne	25	29	81.5	38

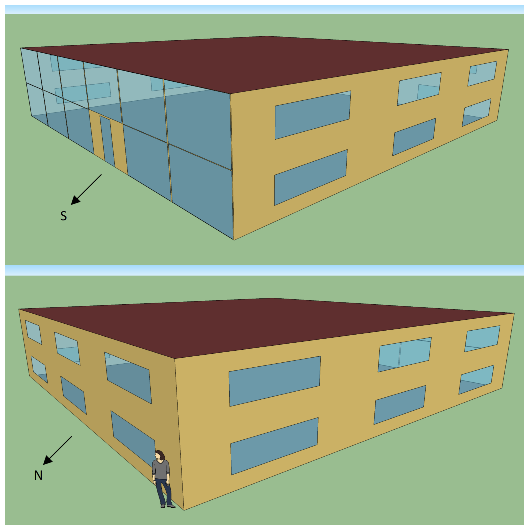
Table 6.6: Window percentage of top solutions.



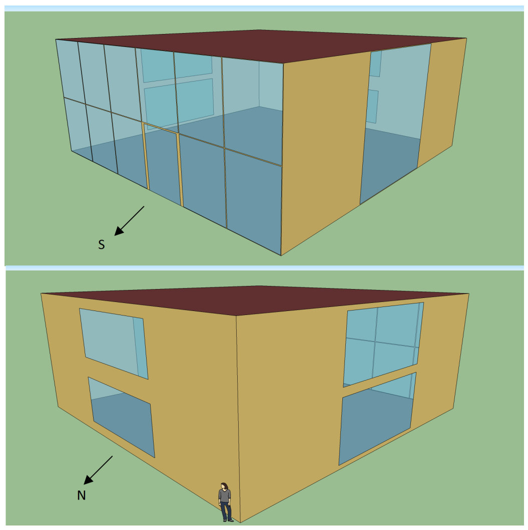
(a) Anchorage, Alaska



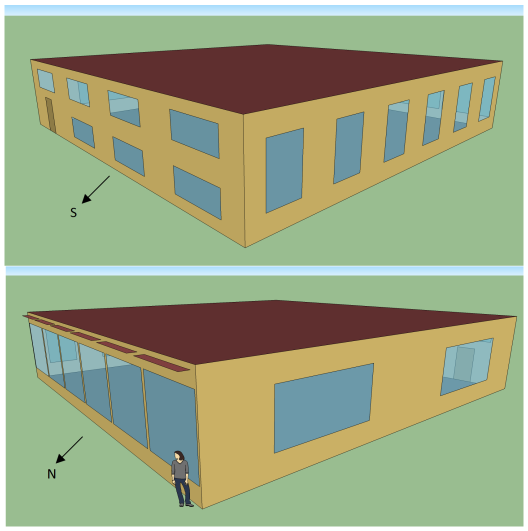
(b) Eldoret, Kenya



(c) Toronto, Canada

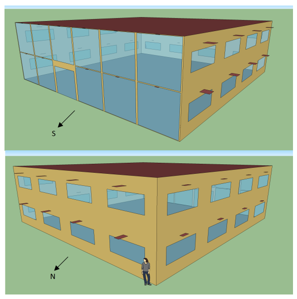


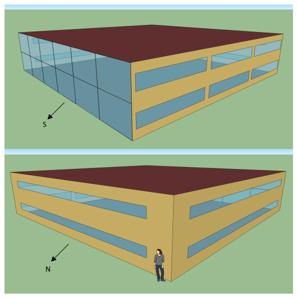
(d) Las Vegas, USA

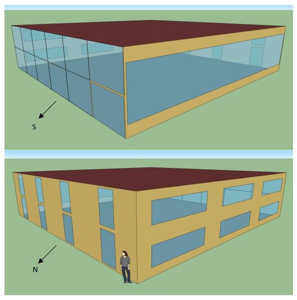


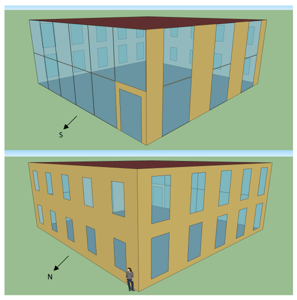
(e) Melbourne, Australia

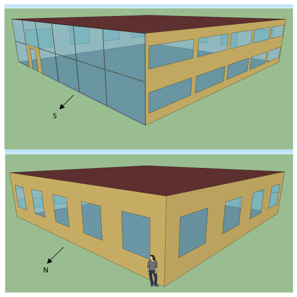
Figure 6.6: Five locations experiment: 3D model of the best solution

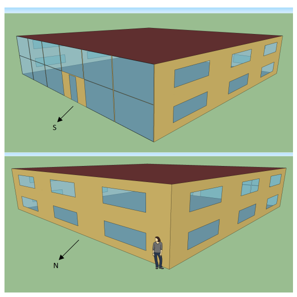

(a) 1

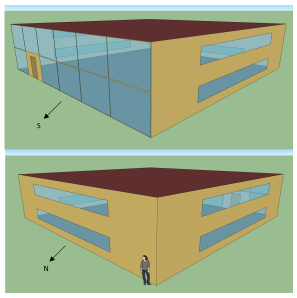

(b) 2

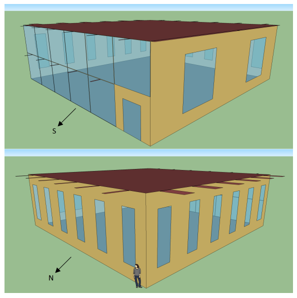

(c) 3

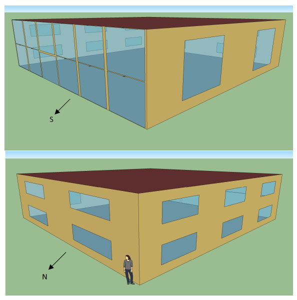

(d) 4


(e) 5


(f) 6


(g) 7


(h) 8


(i) 9

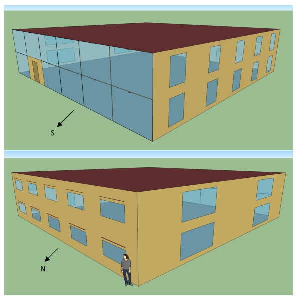
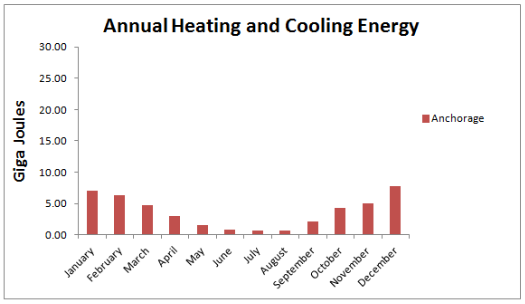
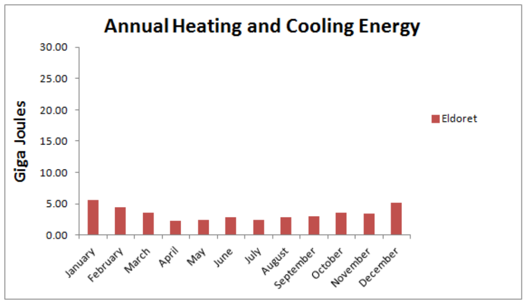

(j) 10

Figure 6.7: Toronto: best model of each trial.

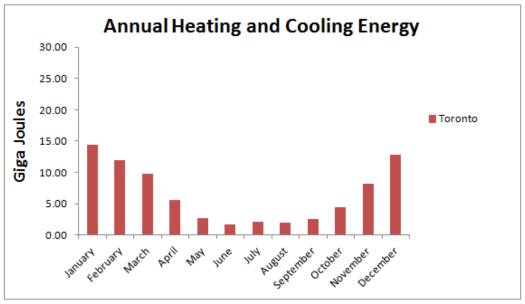
Figure 6.8 displays annual energy consumption for the best model of each city. Anchorage, Toronto, and Las Vegas best models consume energy more in October to March where the weather is colder. Eldoret's best model consumes more energy from December to February. In general, there is not a lot of difference in energy consumption during other months. By comparing Toronto, Anchorage, and Eldoret we can conclude that the need for energy to cool down buildings during summer is high in Eldoret. Las Vegas is the location that consumes the most energy for cooling during summer. Melbourne uses more energy during winter as well. The best GP tree of the Melbourne model and Las Vegas model is given in Appendix C.



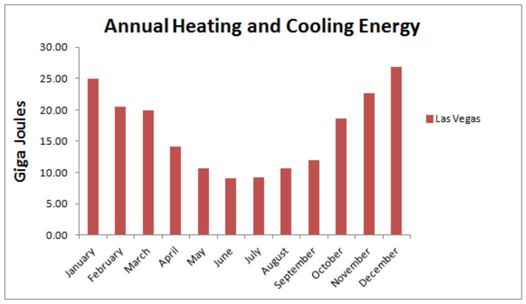
(a) Anchorage, Alaska



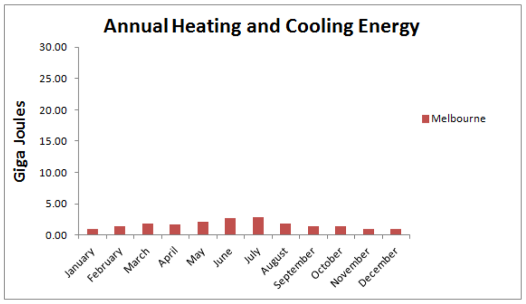
(b) Eldoret, Kenya



(c) Toronto, Canada

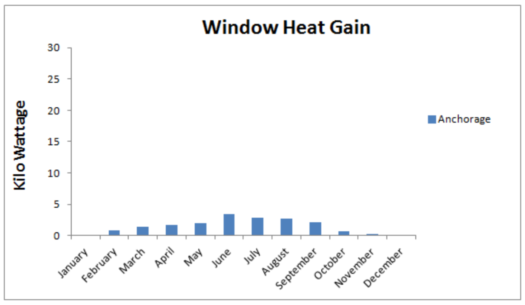


(d) Las Vegas, USA

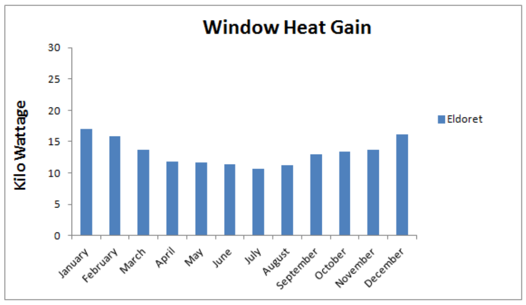


(e) Melbourne, Australia

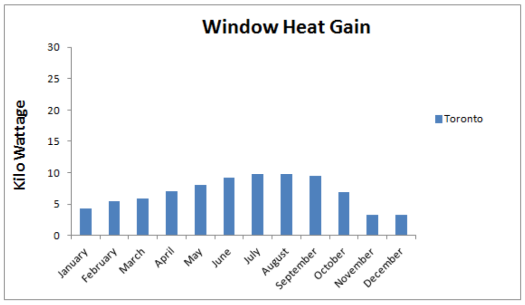
Figure 6.8: Five locations experiment: annual energy consumption



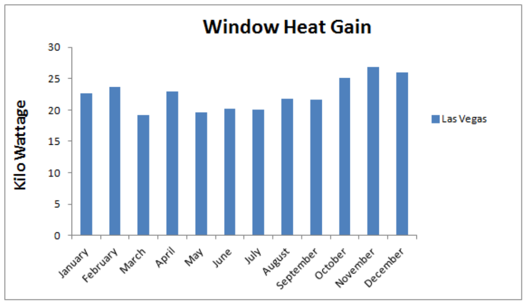
(a) Anchorage, Alaska



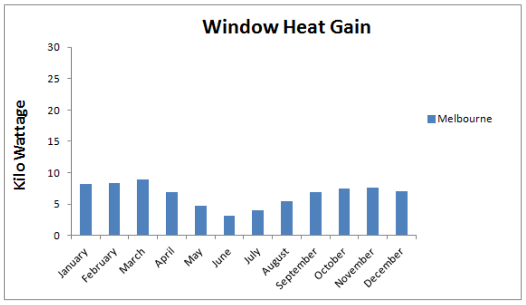
(b) Eldoret, Kenya



(c) Toronto, Canada



(d) Las Vegas, USA



(e) Melbourne, Australia

Figure 6.9: Five locations experiment: window heat gain performance in a year.

Figure 6.9 shows window heat gain for the best models of each location. Las Vegas gains the most heat from windows during the year and Anchorage gains the least heat from windows. On average, Las Vegas gains 23 KW heat from sun monthly. Figure 6.9c, and 6.9e show reverse heat gain pattern for Toronto and Melbourne, and the reason is that Toronto is above the equator and Melbourne is below the equator.

Table 6.7 shows different materials used for the best 10 solutions of each location. In all locations the width and the length of the solutions are maximum except for Anchorage that its size is 18x15. Among all cities, Anchorage has the shortest solutions and Eldoret has the tallest solutions. This shows that the height of the building is relative to the distance of the location to the equator.

In all 50 best solutions of the 5 locations, Floor_2 (hardwood and concrete) and Roof_1 (No mass with thermal resistance 0.65) are selected. Both have the biggest

U-factor which means that they conduct heat easily. All walls of the best 50 solutions are Wall_5 (dense brick, insulation, concrete, and gypsum). Some exceptions have been seen in Anchorage for the south wall, but we should note that almost 100% of the south walls of the best models are windows and the wall type does not affect energy efficiency significantly. Wall_5 is the third best in U-factor perspective. The best wall in U-factor does not conduct a lot of heat from outside to inside, while the third one does. Also, the worst wall in U-factor looses a lot of heat of the inside.

As we mentioned in Section 5.1, both U-factor and solar heat gain coefficient (SHGC) affects the quality of the window. Window_4 has the best U-factor. Window_2 is the second best window in U-factor, but its SHGC is relatively high compare to all double pane windows. All best solutions of all locations except Anchorage selected Window_2 for all sides. Anchorage is close to the North Pole and relatively there is negligible window heat gain for the north side of its models. Therefore, in some cases Window_4 which has the best U-factor is selected to prevent window heat loss.

In Melbourne, Door_1 which has the smallest U-factor is selected in all 10 best solutions. In Eldoret, and Las Vegas, Door_3 (glass door) is selected for all 10 best solutions. Since Door_3 is a glass door, it gains heat and its heat gain coefficient is close to double pane windows. Therefore, although Door_3 has a bad U-factor compare to other doors, it can gain heat. The 10 best solutions of Toronto and Anchorage have selected either Door_1 or Door_3.

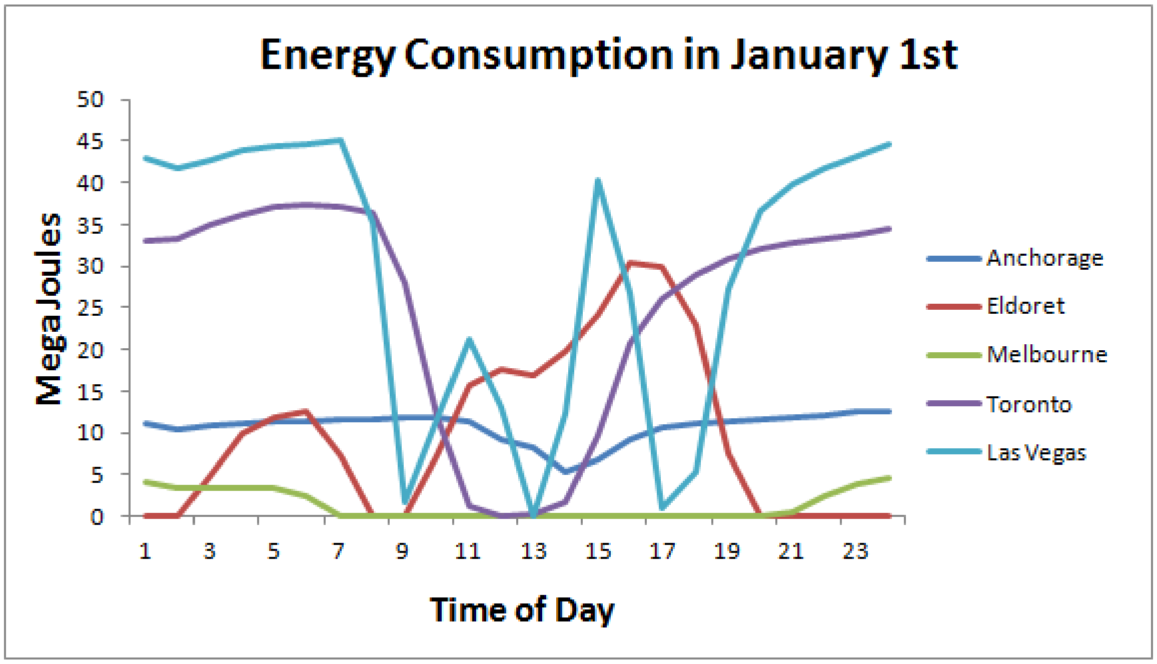
	TO	**AN**	**EL**	**LV**	**ME**
Size	20x20x5	18x15x4.5	20x20x6.5	20x20x6	20x20x5
Floor	2	2	2	2	2
South Wall	5	1,2,5	5	5	5
North Wall	5	5	5	5	5
West Wall	5	5	5	5	5
East Wall	5	5	5	5	5
South Window	2	2	2	2	2
North Window	2	2,4	2	2,4	2
West Window	2	2,4	2	2	2
East Window	2	2,4	2	2	2
Roof	1	1	1	1	1
Door	1,3	1,3	3	3	1

Table 6.7: Material of the best 10 solutions for each location. The number in cells denoted the wall, floor, window, roof, or door number e.g. South Wall = 5 means that Wall_5 is used for the south wall.

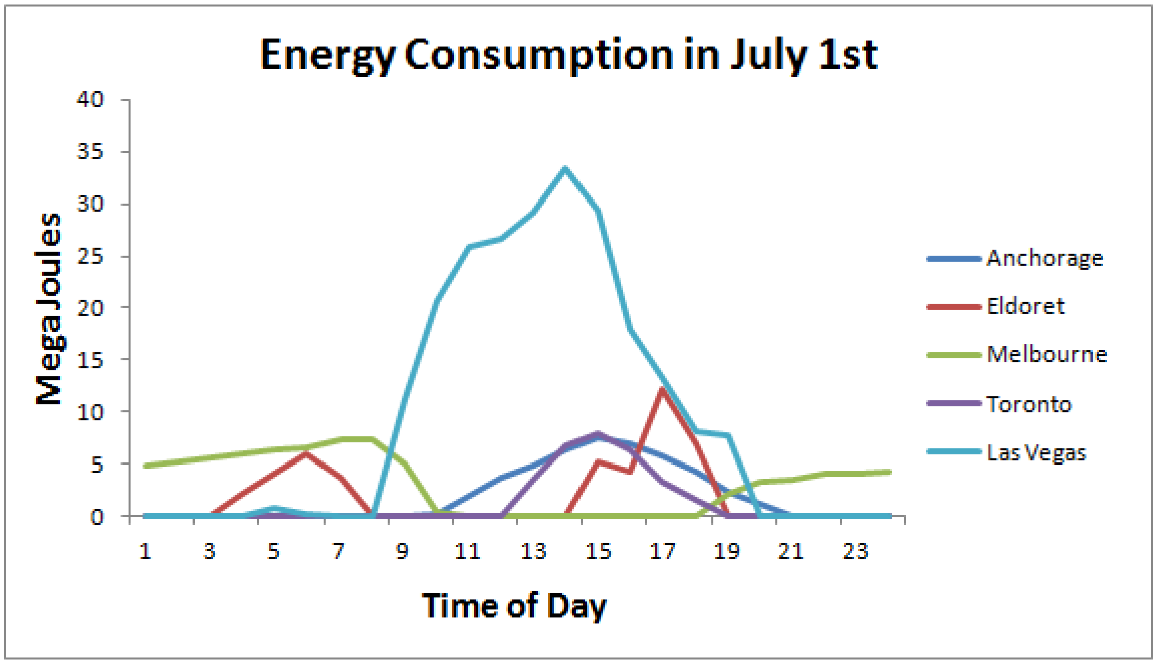
Figure 6.10 represents the energy consumption of the best 5 models in January and July 1st. January 1st and July 1st are arbitrarily chosen as a representative of winter and summer respectively.

In Figure 6.10a, Melbourne has the lowest energy consumption. January is summer in Melbourne and the energy consumption for cooling is low during days. On the other hand, since window heat gain helps heating the house during days, Toronto, Anchorage, and Las Vegas have more energy consumption during nights and early in the morning while window heat gain is zero or negligible. In Eldoret, there is a need to heat up the building from 2 AM to 8 AM. During days, because of at least 25% window on each wall and as winter is not very cold in Eldoret, cooling system cool down the model from 9 AM to 8 PM.

In Figure 6.10b, all locations have relatively low energy usage except Las Vegas. Las Vegas gains a lot of energy by windows and the solutions have to cool down the temperature to keep it below $T_{max} = 24$.



(a) January



(b) July

Figure 6.10: Energy consumption of the best model for each location in January and July 1st.

6.3 Summary

This chapter examined models with the same design elements and objectives on different locations. The objectives were window heat gain in winter, annual energy consumption, and minimum 25% window area.

We chose locations based on their varied geography and weather conditions. Experiments show that Anchorage models have the least window heat gain during winter and Las Vegas models have the best window heat gain during winter. Toronto models have the most energy consumption, while Melbourne has the least energy consumption over a year. Toronto annual energy usage is more than Anchorage because of the cooling energy used in a year. Note that the 25% window area constraint affects all models and in some situations, that can be the reason that annual energy consumption of the models of some locations such as Las Vegas, is high.

Locations in north hemisphere which are Toronto, Anchorage, and Las Vegas, tends to have more windows facing south, while Locations in south hemisphere such as Melbourne tends to have more windows facing north. In fact, this shows how different models for different locations react to daylight. Eldoret is close to the equator and its models have more window area facing east and west.

Model analysis shows us that skylights and roofs other than flat roofs are not energy efficient. Finally, material analysis determines that Wall_5 (dense brick, insulation, concrete, and gypsum), Window_2 (3 mm glass, 13 mm argon, and 3 mm glass), Roof_1 (no mass with thermal resistance 65%), and Floor_2 (concrete, and hardwood) are the selected constructions by GP, since they are the most frequent selected constructions. Based on the location either Door_1 (4 mm wood) or Door_3 (single layer 3 mm grey glass) is preferred.

Chapter 7

Multi-floor Experiments

All previous experiments considered single floor models. In this chapter, the impact of multi-floor design on materials, energy consumption, and window heat gain is examined. All models to be studied have five floors. The geographic location used for all experiments is Toronto.

7.1 Experimental Setup

Table 5.1 presents the GP parameters used, except that the generation number is increased to 80. Table 6.2 shows the design parameters used, and Table 5.3 and 5.4 present all materials used. Table 7.1 presents the design functions and terminals. The two new functions are Add Floor and Add Root and the rest are as same as Table 6.3. Here, Add Root function accepts five floors.

GP Functions	Add Root, Add Cube, Add Floor, First Floor, Add Door Grid, Add Grid, Add Door, Add Window, Add Window Overhang, Add Empty Grid, Add Simple Roof, Add Skylight, Add Gabled Roof, Add Gabled Roof2, GP mathematical functions
GP Terminals	ERC, Int_ERC

Table 7.1: Design functions: these functions and terminals are used in all experiments of this section.

7.2 Window Heat Gain in Winter, Annual Heating and Cooling Energy, and at Least 25% Window Area for Each Floor

The objectives of this experiment are identical to the objectives in Section 5.2.2. Window heat gain, annual energy consumption, and at least 25% window area are the objectives of this experiment. The only difference is that 5 floor models are used here.

Figure 7.1, 7.2, and 7.3 depict annual energy consumption, window heat gain, and SOE over generations respectively. Both population average and the best average is presented. A comparison between Figure 7.1 and 6.2 shows that the annual energy consumption for a five floor model is less than a single floor model multiplies by five. The reason is that the floors conduct heat between each other. In addition, five single floor models lose heat from roofs more than a five floor model. Comparing Figures 7.2 and 6.3 reveals that a single floor model's heat gain multiply by five is more than a five floor model's heat gain. Larger windows facing south in single floor models is the reason of this fact. Figure 7.3 shows that SOE converges to zero after 30 generations, while in single floor models SOE converges to zero after 15 generations. The reason is that the five floor models are more complex than single floor models and GP have more difficulties in finding good solutions.

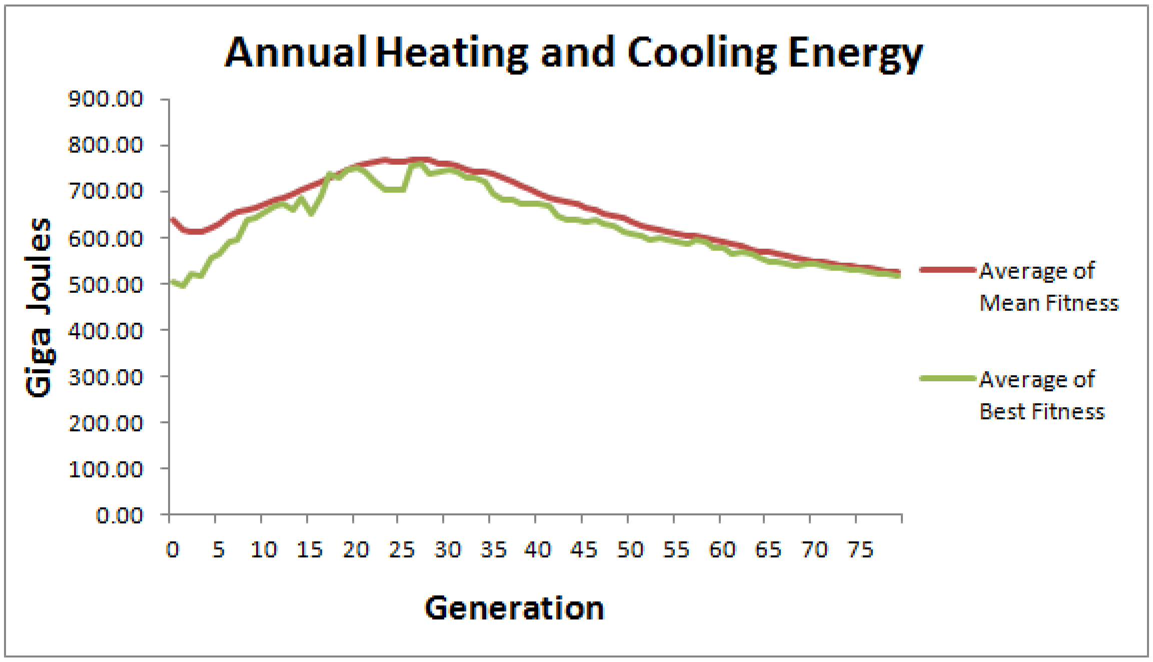


Figure 7.1: Population average annual energy performance plot.

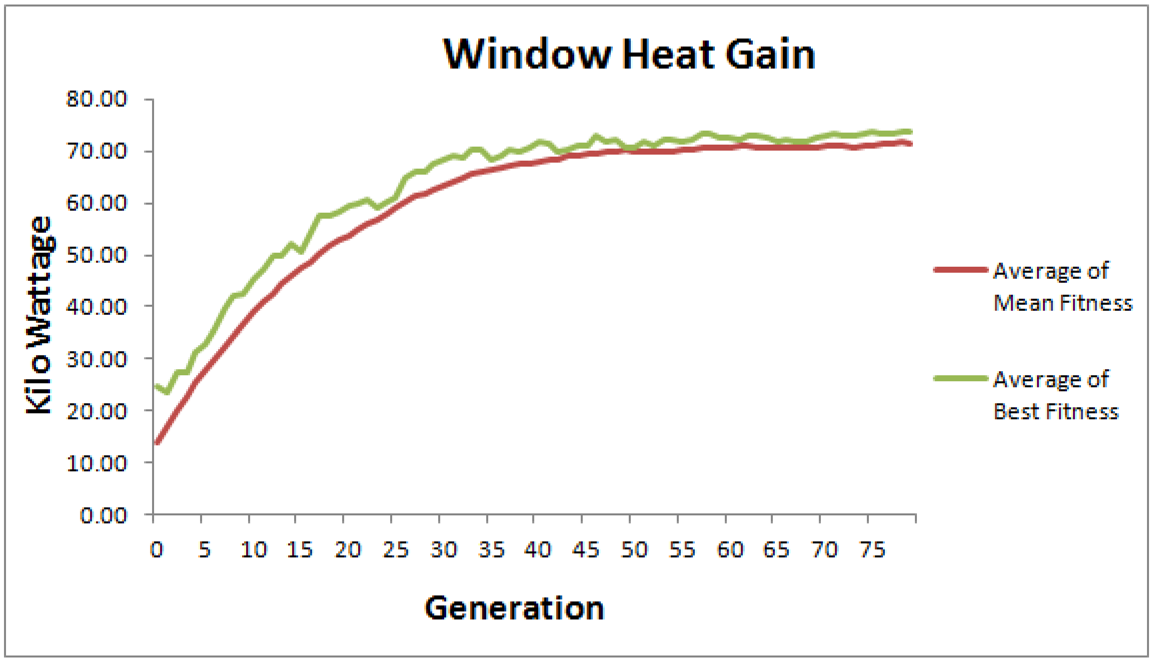


Figure 7.2: Population average window heat gain performance plot.

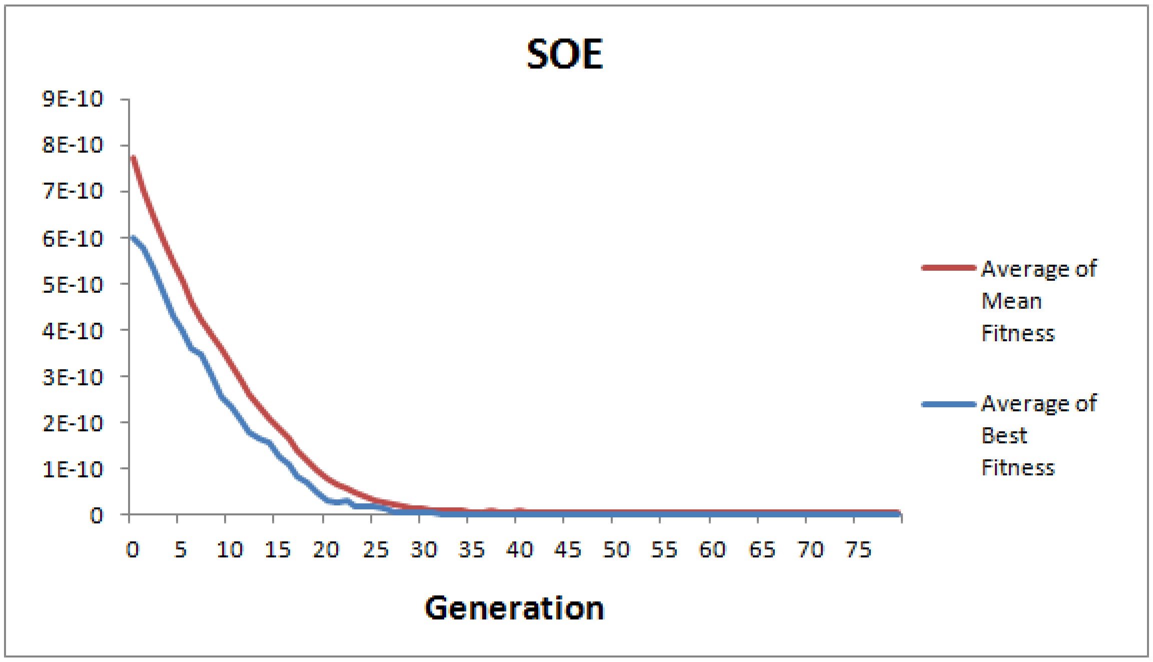
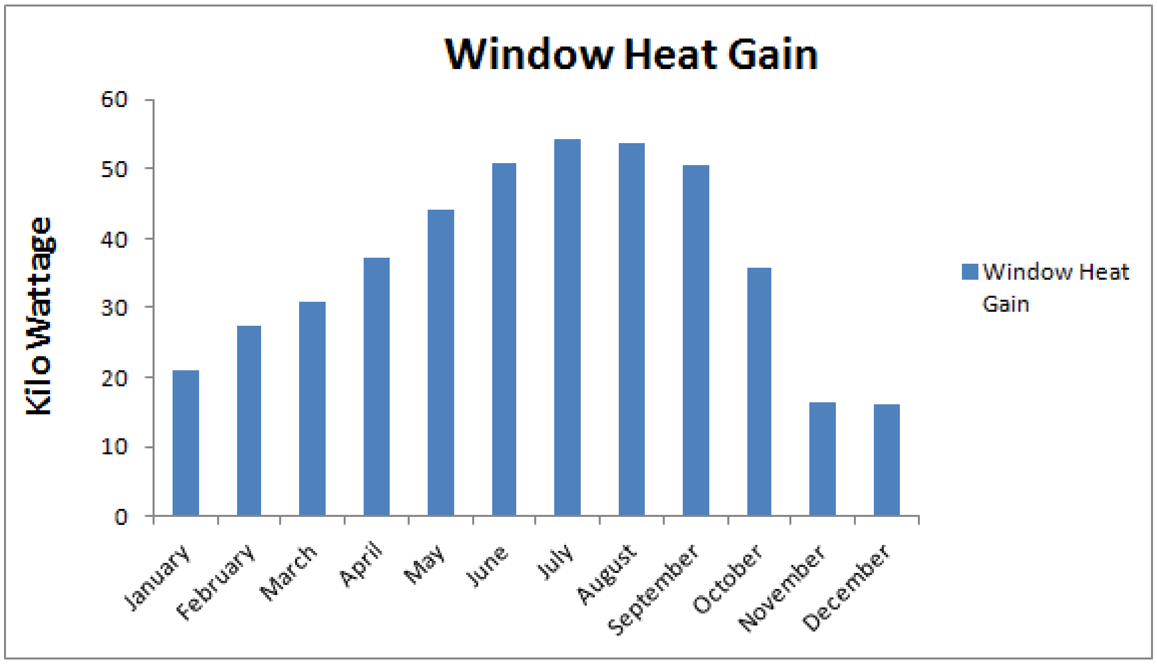
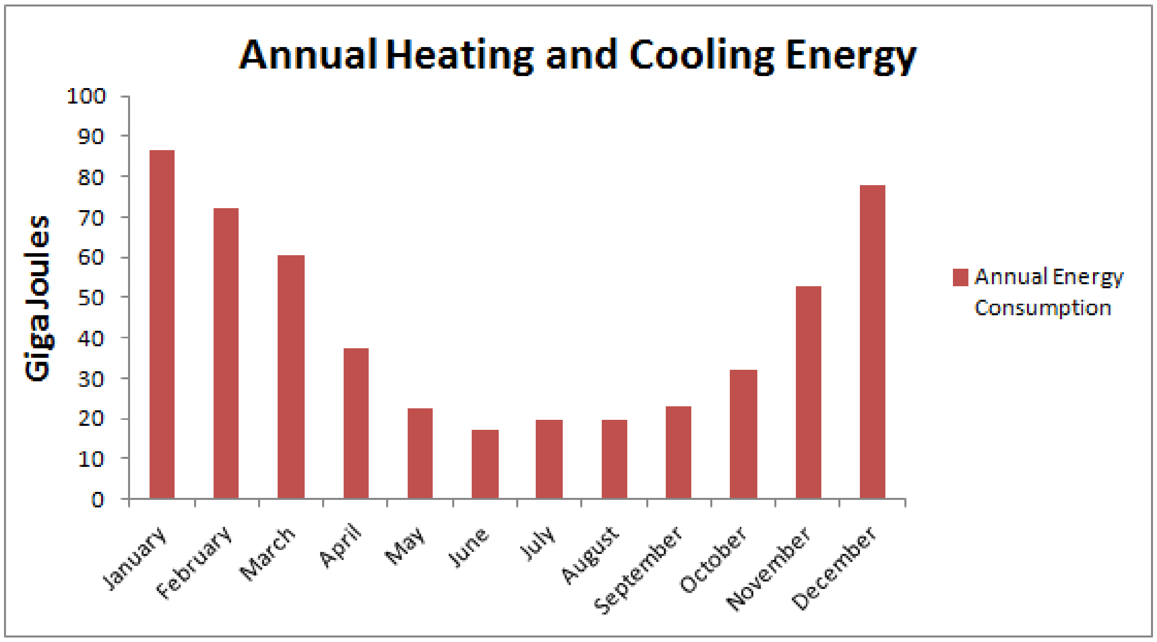


Figure 7.3: Population average window area sum of errors.



(a) Window heat gain



(b) Annual energy consumption

Figure 7.4: Annual energy consumption and window heat gain of the best model of all trials.

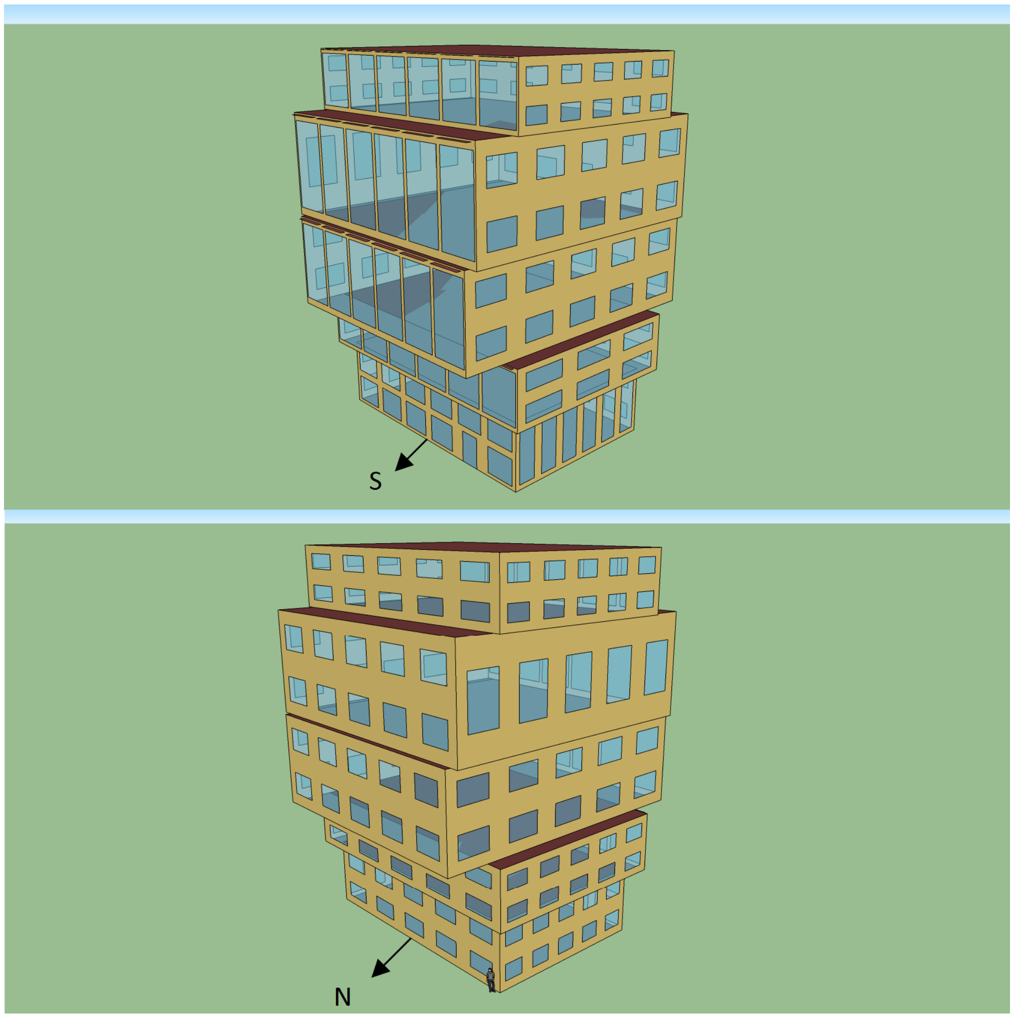


Figure 7.5: The best model of 10 trials.

Figure 7.4 depicts annual energy consumption and window heat gain of the best model of all trials. The best solution of each trial is taken and the best evolved model is selected by NRS from them.

Figure 7.5 illustrates the best model of all runs. This model has 75% window area facing south and approximately 30% window area facing other directions.

Table 7.2 shows the result of a material analysis. The material analysis shows that the most selected wall of the 10 best solutions is Wall_5 (dense brick, insulation, concrete, and gypsum). Wall_1 and Wall_2 are the best and the second best wall in case of U-factor in our system. All windows of the best 10 solutions are double pane. 75% windows of the best 10 solutions are Window_2 (double pane with argon). This window is the second best in U-factor, and the best in SHGC in all double pane windows.Window_1 and Window_4 have the third best and the best U-factor. Roof_1 (no mass with thermal resistance 0.65) with the biggest U-factor is selected 75% of the time in the best 10 models. Floor_2 with the second best U-factor is selected 85% of the time. Door_3 (glass door) with the biggest U-factor and Door_1 (4 mm wood) with the best U-factor are selected equally.

Construction	**% usage**
Wall_1	15%
Wall_2	15%
Wall_5	70%
Window_1	12%
Window_2	75%
Window_4	12%
Roof_1	75%
Roof_2	15%
Roof_3	10%
Floor_1	15%
Floor_2	85%
Door_1	50%
Door_3	50%

Table 7.2: Material analysis of the best 10 solutions.

7.3 Window Heat Gain in Winter, Annual Heating and Cooling Energy, and 35% Window Area

This section's objectives are similar to the previous section's objectives with one difference: window area must be exactly 35% for each wall. In other words, SOE is calculated by the following formula:

$$SOE = \sum_{i \in \{walls\}} (0.35 - \frac{window\ area\ of\ the\ wall_i}{wall_i\ area})^2$$

Figure 7.6, 7.7, and 7.8 displays annual energy consumption, window heat gain, and SOE over generations respectively. Comparing Figure 7.6, and 7.7 with Figure 7.1, and 7.2 show that smaller windows cause lower annual energy consumption and lower window heat gain. In addition, Figure 7.8 determines that the SOE converges to zero after 70 generations while SOE in Figure 7.3 converges to zero after 30 generations. Therefore, making good solutions with exactly 35% window area is harder than making good solutions with at least 25% window area.

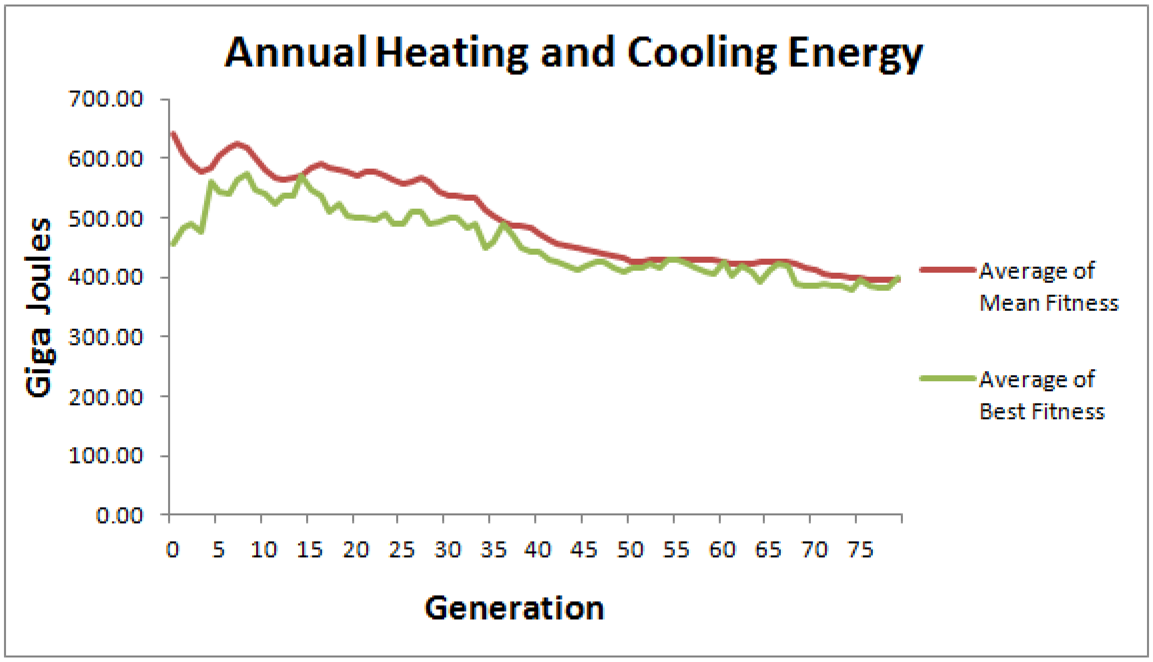


Figure 7.6: Population average annual energy performance plot.

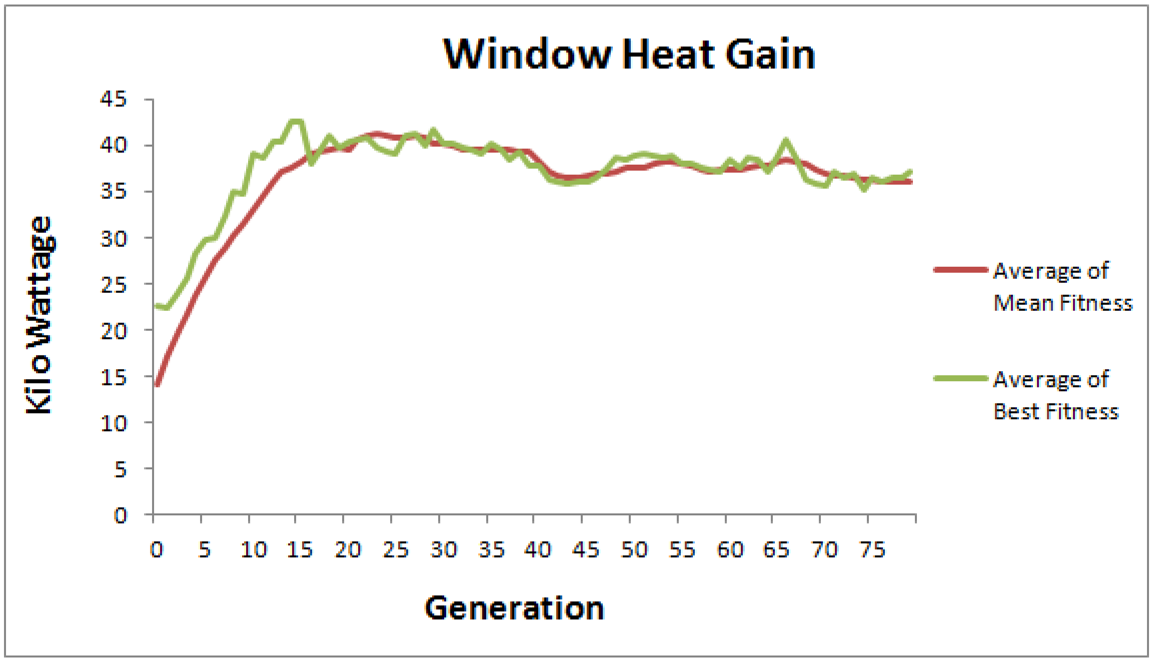


Figure 7.7: Population average window heat gain performance plot.

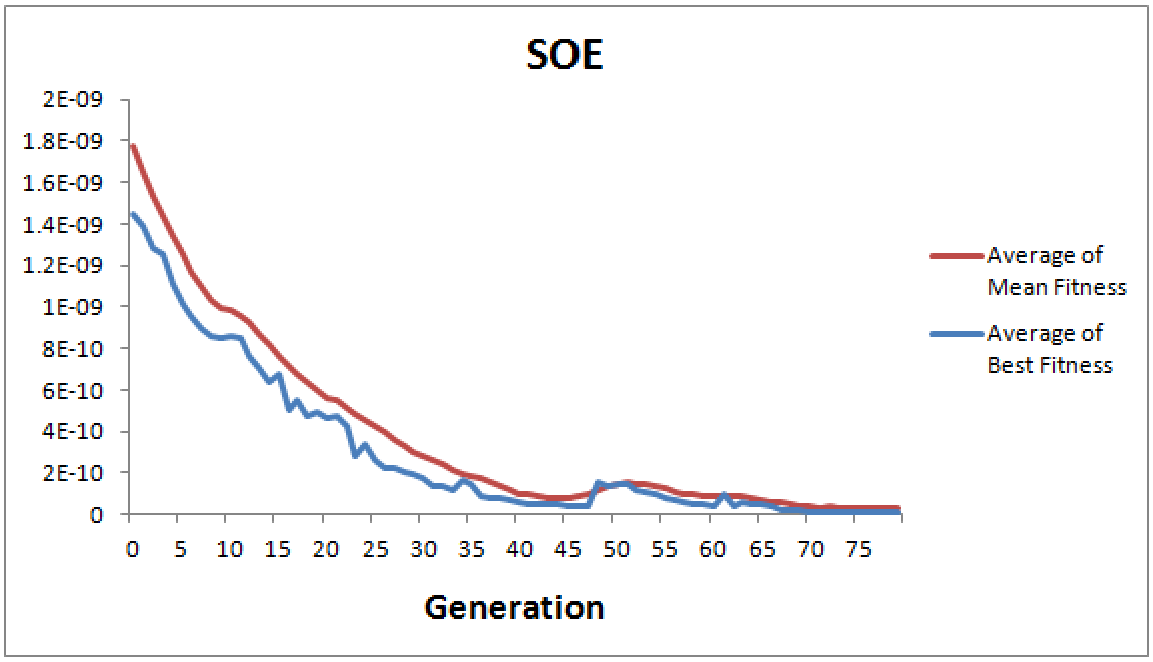
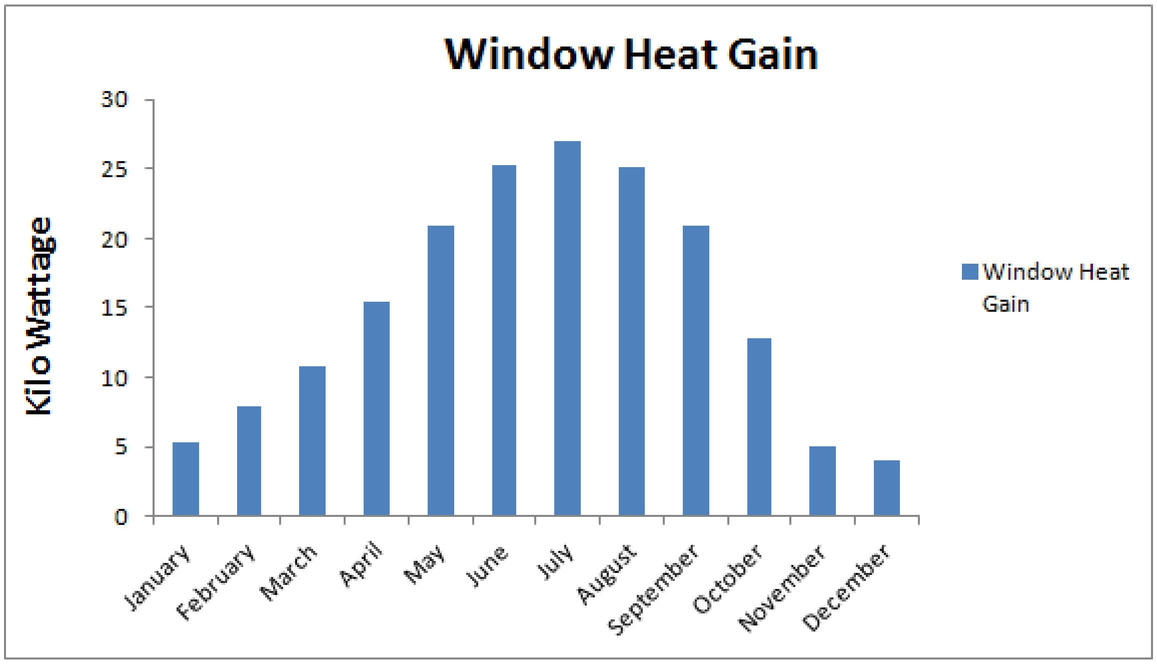
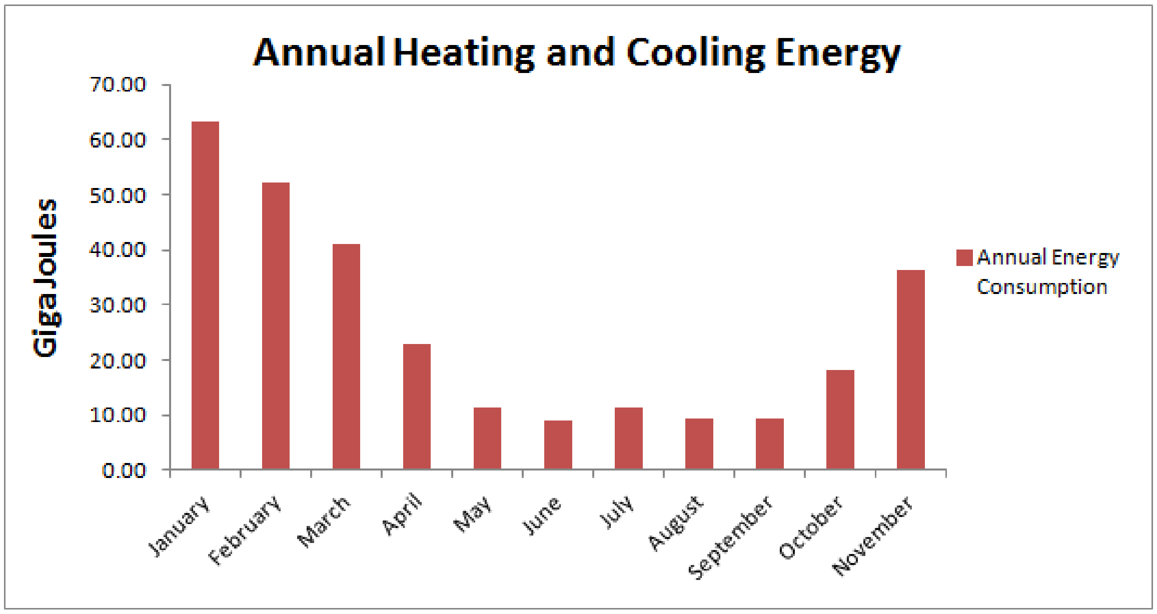


Figure 7.8: Population average window area sum of errors.

Figure 7.9 depicts annual energy consumption and window heat gain of the best model of all runs. The best model is selected by applying NRS on the best 10 solutions of the trials. Comparing Figure 7.9 and 7.4 determines that the best model of this experiment gains less heat during winter and uses less energy for cooling and heating during a year compare to the best model of the previous experiment.



(a) Window heat gain



(b) Annual energy consumption

Figure 7.9: Annual energy consumption and window heat gain of the best model of all trials.

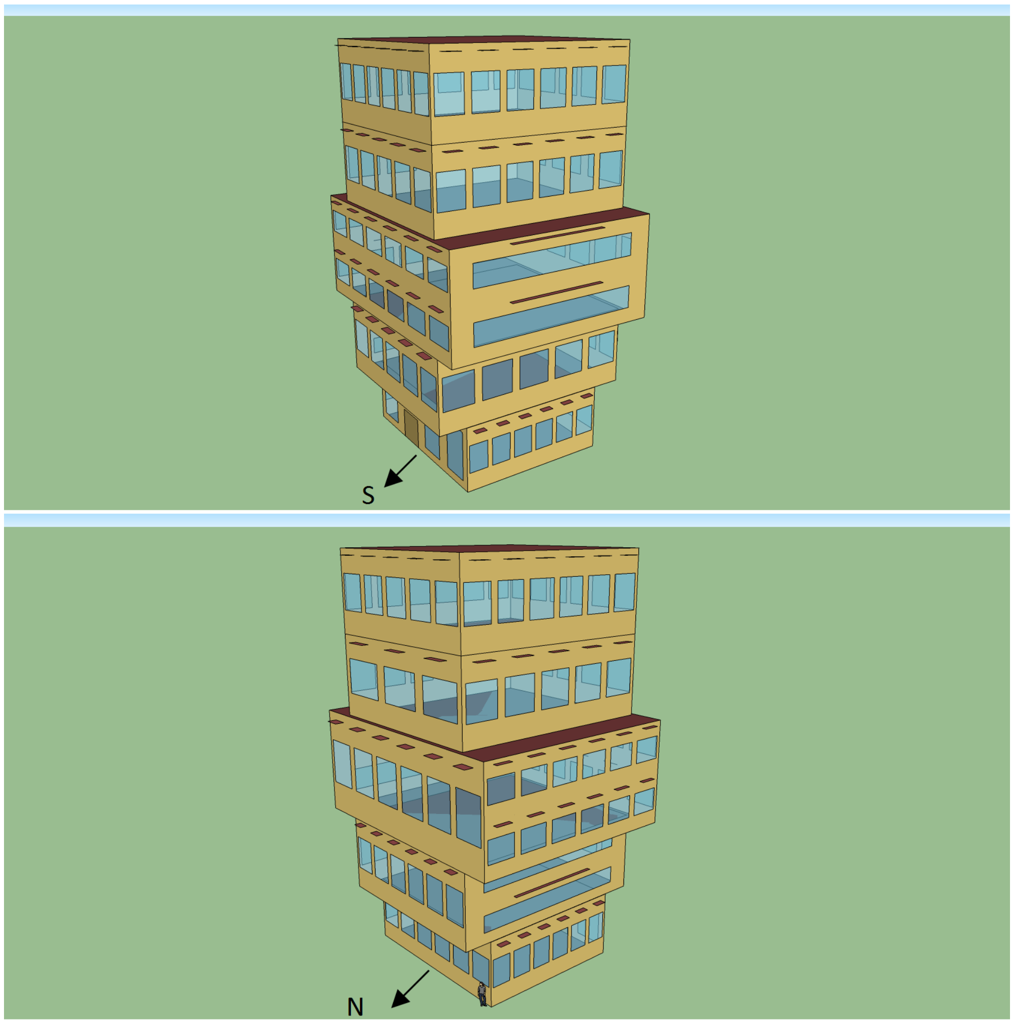


Figure 7.10: The best model of 10 trials.

Figure 7.10 illustrates the best model of all runs. This model has approximately 34.25% window area on each wall which is close to 35%. Figure 7.11 illustrates a real model similar to our model in some aspects. This building is designed by Statoil company [33] and it is located in Norway.

The result of the material analysis is shown in Table 7.3. Wall_5 (dense brick, insulation, concrete, and gypsum) is the most selected wall among all 5 different walls. This wall is the third best wall in U-factor. Windows in the 10 best models are double pane. Although the most selected window is Window_2 (double pane with argon), this experiment was a hard one for GP to determine the best window. Floor_1 with the best U-factor and Floor_2 with the second best U-factor are selected 50%. Roof_1 (no mass with thermal resistance 0.65) with the biggest U-factor is the most selected roof. Door_3 (glass door) with the biggest U-factor and Door_1 (4 mm wood) with the best U-factor are selected 40%. The result of the material analysis shows that this experiment is a difficult experiment for GP to determine the best materials in 80 generations. Perhaps by increasing the number of generations, materials for walls, windows, etc. converges to a specific kind.

Construction	**% usage**
Wall_1	18%
Wall_2	15%
Wall_5	67%
Window_1	30%
Window_2	41%
Window_4	28%
Roof_1	68%
Roof_2	18%
Roof_3	14%
Floor_1	50%
Floor_2	50%
Door_1	40%
Door_2	20%
Door_3	40%

Table 7.3: Material analysis of the best 10 solutions.

Figure 7.11: A real building: Statoil [33].

7.4 Window Heat Gain in Winter, Annual Heating and Cooling Energy, and Volume Constraints

In this section, we add volume constraints as objectives alongside the two energy based objectives used before: window heat gain in winter, and annual heating and cooling energy. The first volume constraint is that the (i+1)-th floor must be at least 15% smaller than the i-th floor where i $\in \{1, 2, 3, 4\}$. If the (i+1)-th floor is not 15% smaller than the i-th floor, the difference is considered as error. Sum of the squared errors is called SOE and is calculated by the following formula:

$$SOE = \sum_{i=1}^{4} (Volume_i * (1 - 0.15) - Volume_{i+1})^2$$

where i $\in \{i | Volume_i * 0.85 < Volume_{i+1}\}$.

The second volume constraint is the total volume of the building. The minimum possible volume of this experiment is:

$$\begin{aligned} Minimum\,Volume &= number\,of\,floors * minimum\,width * \\ &\quad minimum\,length * \; minimum\,height \\ &= 5 * 10 * 10 * 4 \\ &= 2000\,m^3 \end{aligned}$$

and the maximum possible volume is:

$$\begin{aligned} Maximum\,Volume &= number\,of\,floors * maximum\,width * \\ &\quad maximum\,length * \; maximum\,height \\ &= 5 * 20 * 20 * 8 \\ &= 16000\,m^3 \end{aligned}$$

For this experiment, the target volume is 10000 m^3. The absolute value of the difference of the total volume with the target volume is considered as error. The volume error is calculated by the following formula:

$$Volume_e = |10000 - \sum_{i=1}^{5} Volume_i|$$

Figure 7.12, 7.13, 7.14, and 7.15 depict annual energy consumption, window heat gain, SOE, and volume error over generations. All the best models of this experiment have windows facing south and most of them have no windows on the other walls. Window area for the south wall is approximately 70% on average. Comparison between Figure 7.13 and 7.2 from Section 7.2 shows that the window heat gain during winter for both experiments are roughly the same. Therefore, we can conclude that the most window heat gain is gained from the windows on the south wall. In addition, by considering Figure 7.12 and 7.1, we can conclude that having windows on other sides cause more demand for cooling and heating during a year. Figure 7.14 tells us that floors are approximately 15% smaller from the floor underneath after generation 40. Also, Figure 7.15 points that the volume of the models of the last generation are almost 10000 m^3.

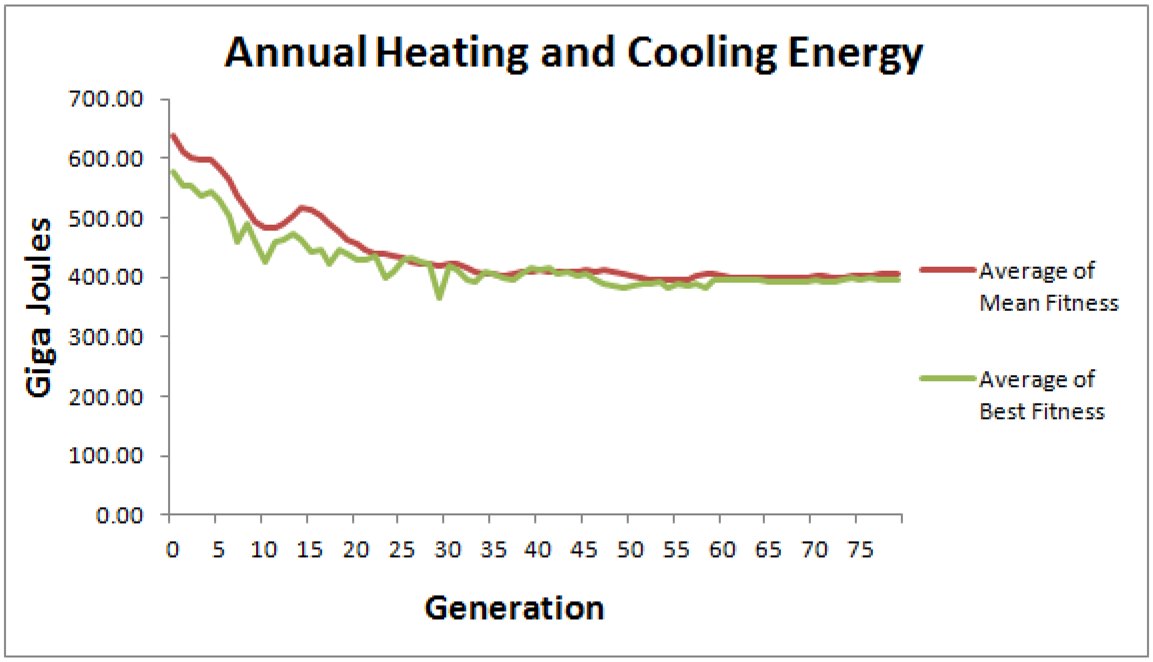


Figure 7.12: Population average annual energy performance plot.

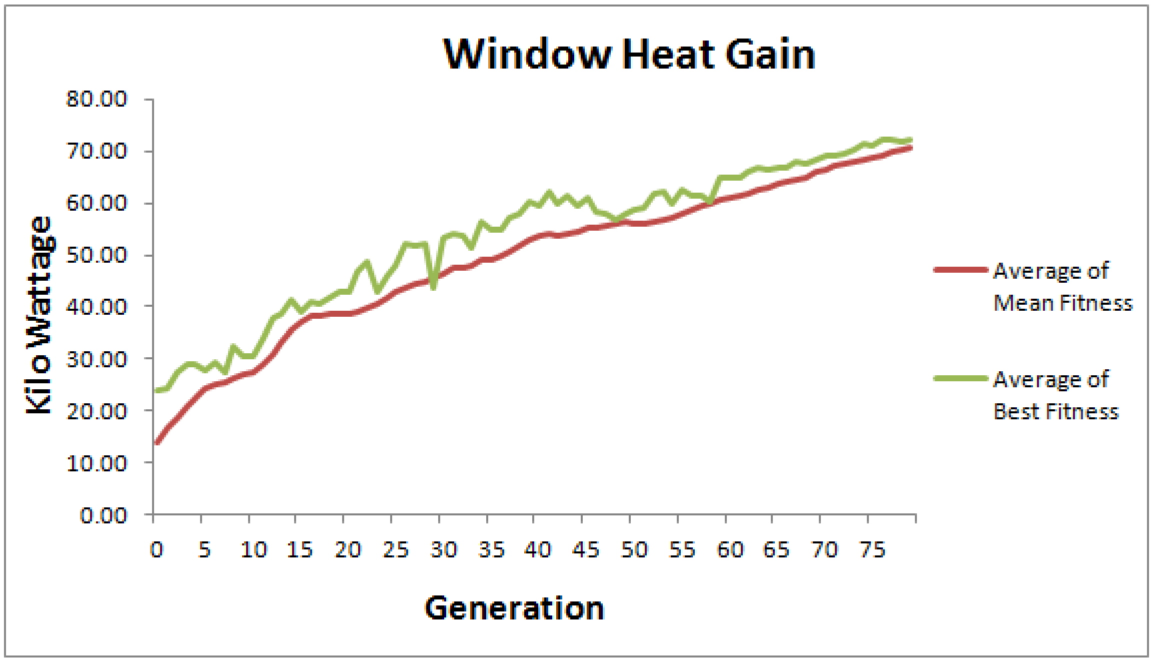


Figure 7.13: Population average window heat gain performance plot.

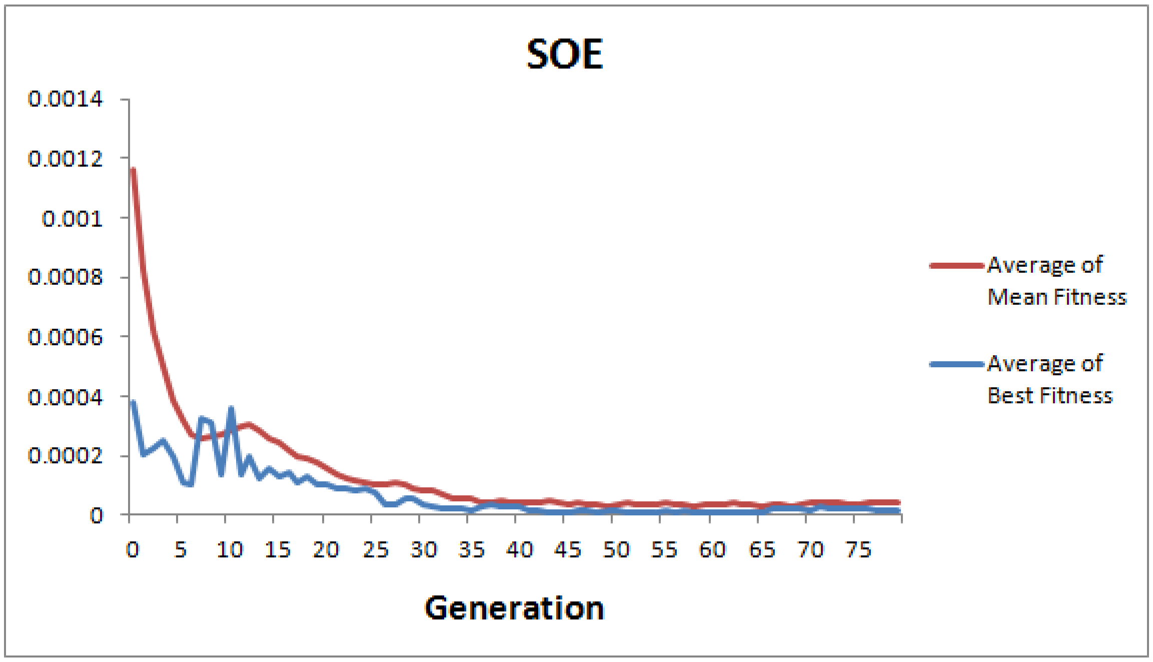


Figure 7.14: Population volume sum of errors.

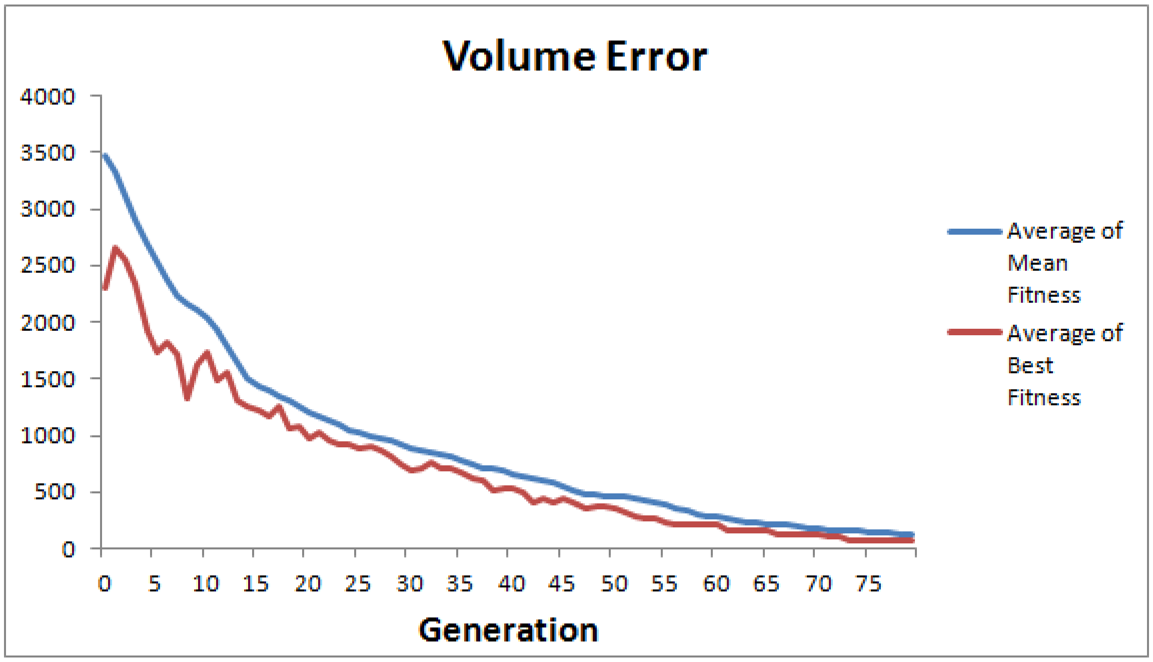
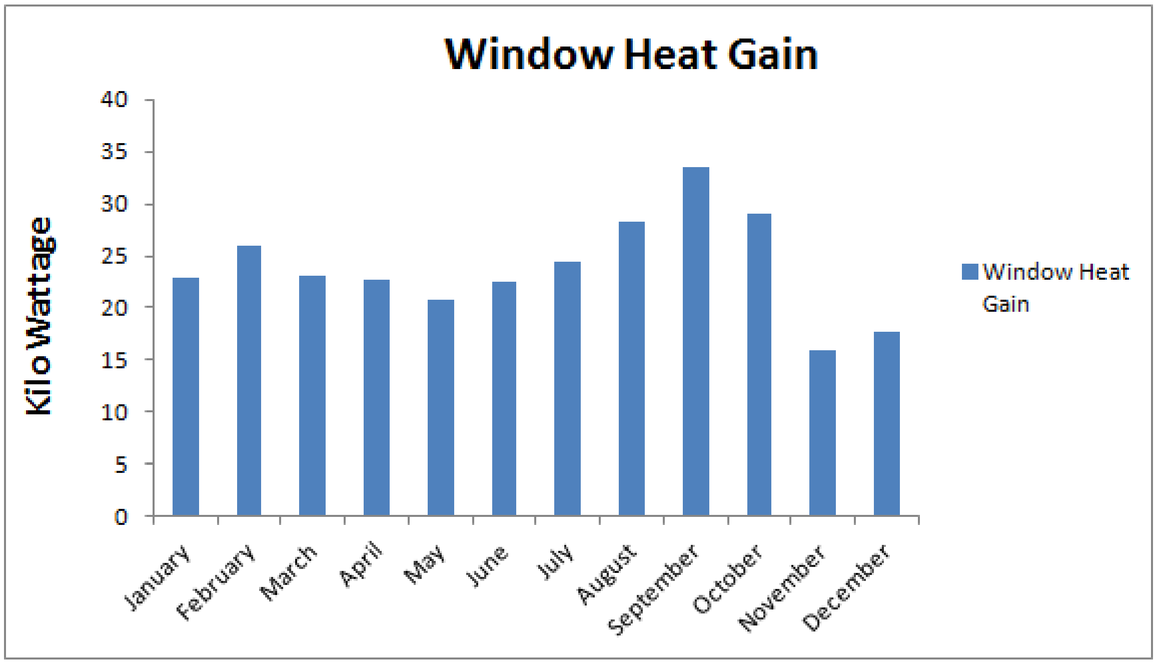
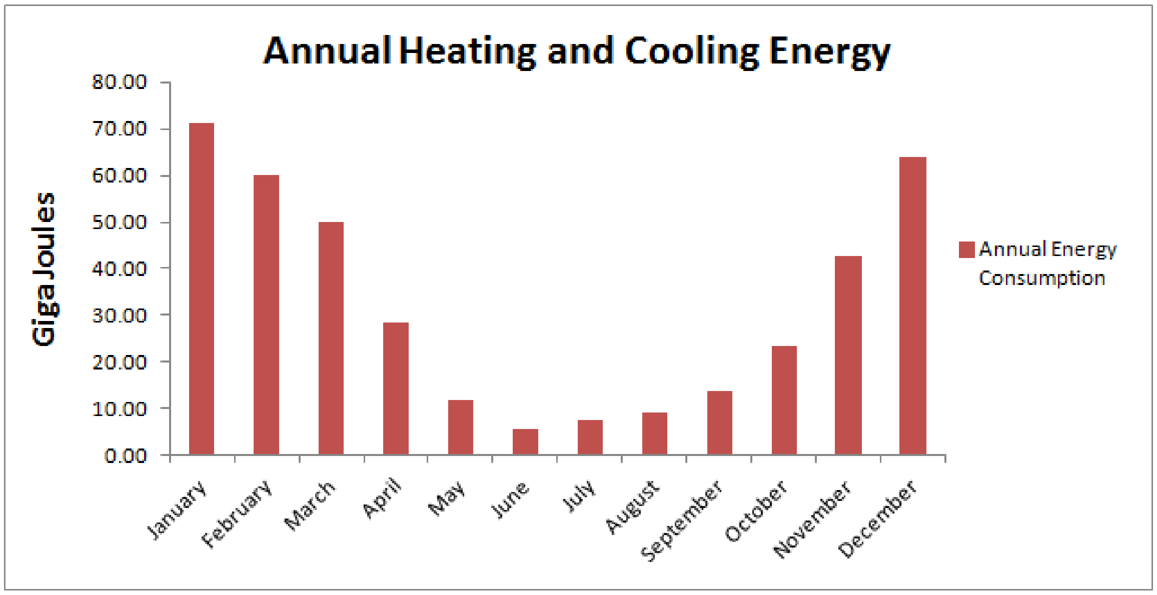


Figure 7.15: Population average volume error.

Figure 7.16 represents annual energy usage and window heat gain of the best model of all trials. Because of having larger windows in the south in this experiment, the window heat gain of this experiment is more than the window heat gain in previous experiment which is presented in Figure 7.7. In addition, since in Section 7.2 there was a constraint of having at least 25% window on each side, the window heat gain of that experiment is more than found here. A comparison between Figure 7.9b and 7.16b also shows that more window area causes more need for energy.



(a) Window heat gain



(b) Annual energy consumption

Figure 7.16: Annual energy consumption and window heat gain of the best model of all trials.

Figure 7.17 illustrates the best model of all runs. This model has 71% window area on south wall and 0% window area on other walls. All floors are at least 15% smaller than the floor underneath.

Table 7.4 shows material usage of the best 10 solutions. Percentages shows the percentage of the use of a specific material in the best 10 solutions. Wall_5 (dense brick, insulation, concrete, and gypsum) is the most selected wall among all 5 different walls. 85% of the walls of the 10 best solutions is Wall_5. Wall_5 is the third best wall in U-factor. Selected windows in all best 10 solutions are double pane windows. Window_2 (double pane with argon) is the second best window in U-factor and the best in SHGC among all double pane windows. Floor_2 is the frequent selected floor. Roof_1 (no mass with thermal resistance 0.65) with the biggest U-factor is the most selected roof. Door_3 (glass door) with the biggest U-factor is selected the most.

Construction	**% usage**
Wall_1	13%
Wall_2	3%
Wall_5	84%
Window_1	15%
Window_2	85%
Roof_1	80%
Roof_3	20%
Floor_1	30%
Floor_2	70%
Door_1	30%
Door_3	70%

Table 7.4: Material analysis of the best 10 solutions.

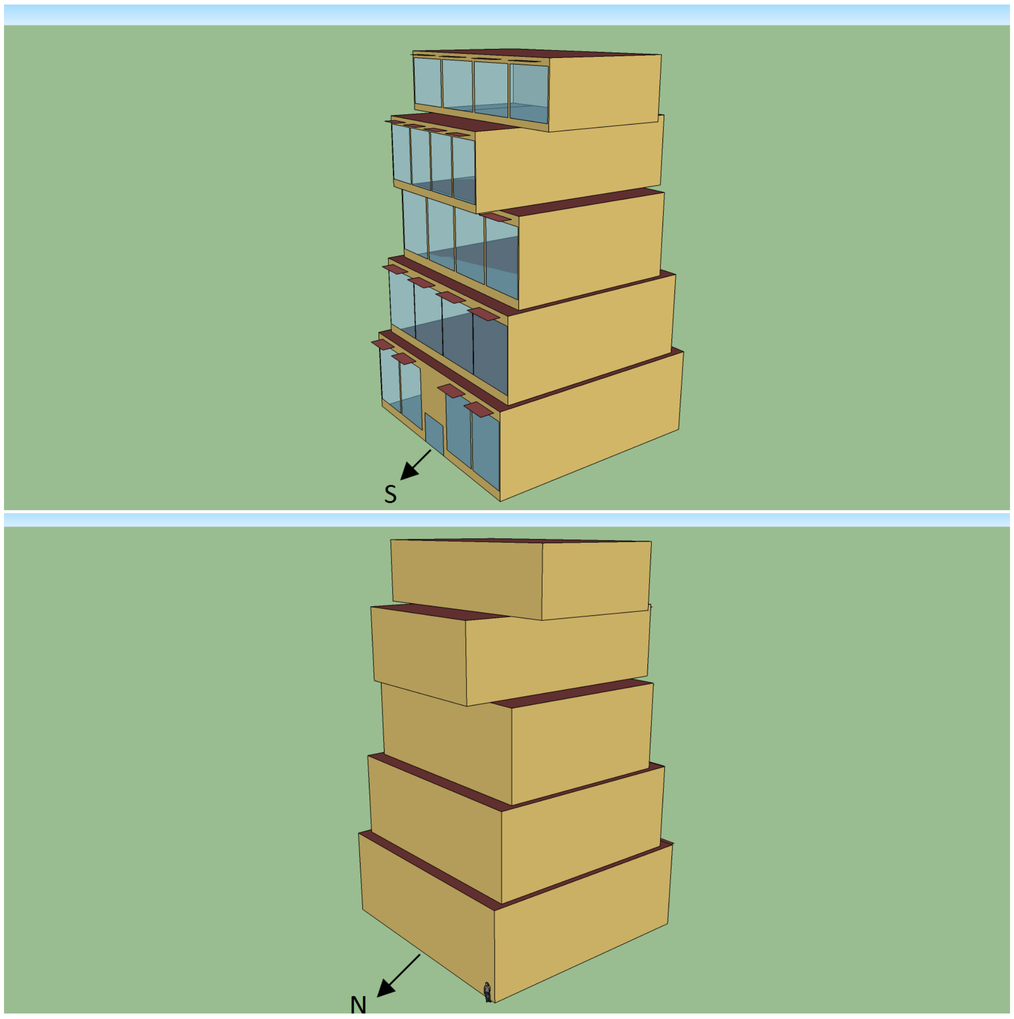


Figure 7.17: The best model of 10 trials.

7.5 Window Heat Gain in Winter, Annual Heating and Cooling Energy, Window Area Constraints, and Volume Constraints

This section's objectives are a union of objectives of Sections 7.3 and 7.4:

1. Window heat gain.
2. Annual energy consumption.
3. Each wall has to have exactly 35% window area. The sum of squared error of this objective is called window SOE.

4. Each floor has to be at least 15% smaller than the floor underneath. The sum of squared error of this objective is called volume SOE.

5. The sum of the volumes of the five floors has to be exactly 10000 m^3. The difference of 10000 m^3 and the volume of the model is called volume error.

As this experiment is the most complex experiment we have carried out in this thesis, the maximum generations is increased to 100.

Figure 7.18, 7.19, 7.20, 7.21, and 7.22 show annual energy consumption, window heat gain, window SOE, volume SOE, and volume error performance plots.

The 10 best solutions have almost exactly 35% window area on each wall. Firstly, a comparison of Figure 7.18 with Figure 7.12 from Section 7.3 and Figure 7.6 from Section 7.4, shows that on average the solutions of this experiment need more energy for cooling and heating. The solutions have larger sizes than solutions in Section 7.3. In addition, the solutions have windows on all sides of the building. Also, the window area facing south is reduced to 35% window area in this experiment. Therefore, there is more window heat loss and more energy is needed for warmth. Secondly, the comparison of Figure 7.19 with Figure 7.7 shows that the window heat gain of this experiment is higher. Both experiments have a goal to have exactly 35% window area, but models found here are larger than the models of that experiment. Furthermore, according to Figure 7.19 and 7.13, window heat gain of the solutions of this experiment is less than the experiment of Section 7.4, since window area facing south for models of this experiment is less than the window area of the other experiment. Thirdly, the comparison of the Figure 7.20 with Figure 7.8 from Section 7.3 shows that this experiment is more complex than the experiment of Section 7.3, because in this experiment even after 100 generations window SOE does not converge to zero, while window SOE in Section 7.3 converges to zero after 75 generations. Fourthly, by comparing Figure 7.21 with Figure 7.14, we can conclude that this experiment has more difficulty in finding models which have floors 15% smaller than the floor underneath. Finally, a comparison between Figure 7.22 and 7.15 determines that the solutions of this experiment have less volume error compared to the solutions of the experiment of Section 7.4.

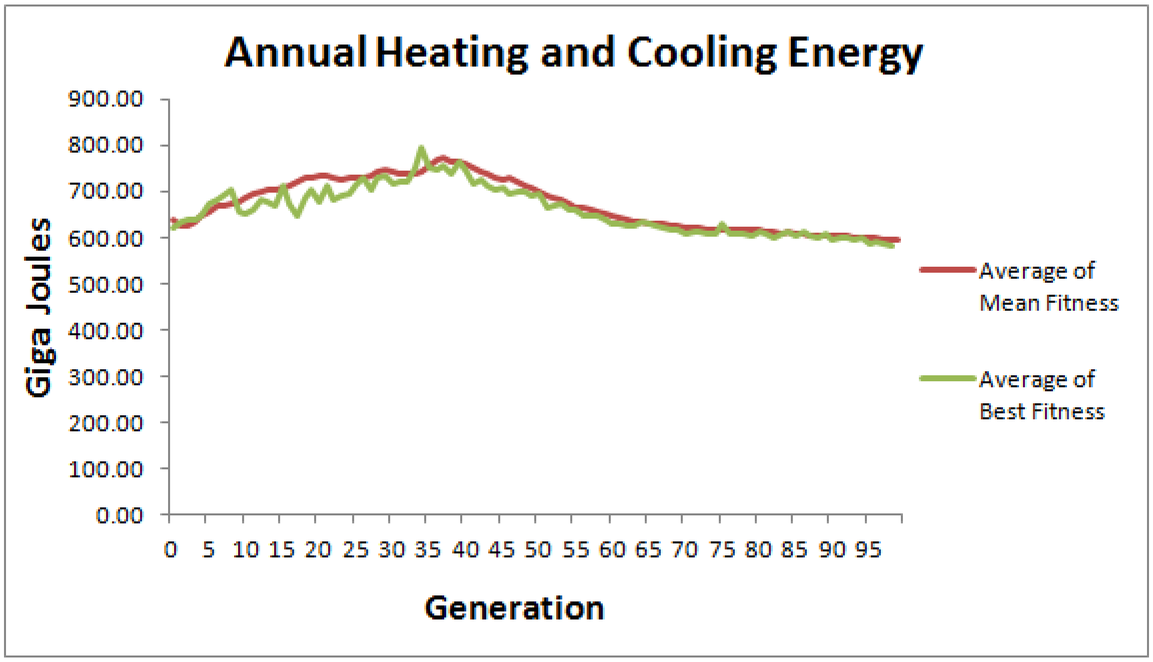


Figure 7.18: Population average annual energy performance plot.

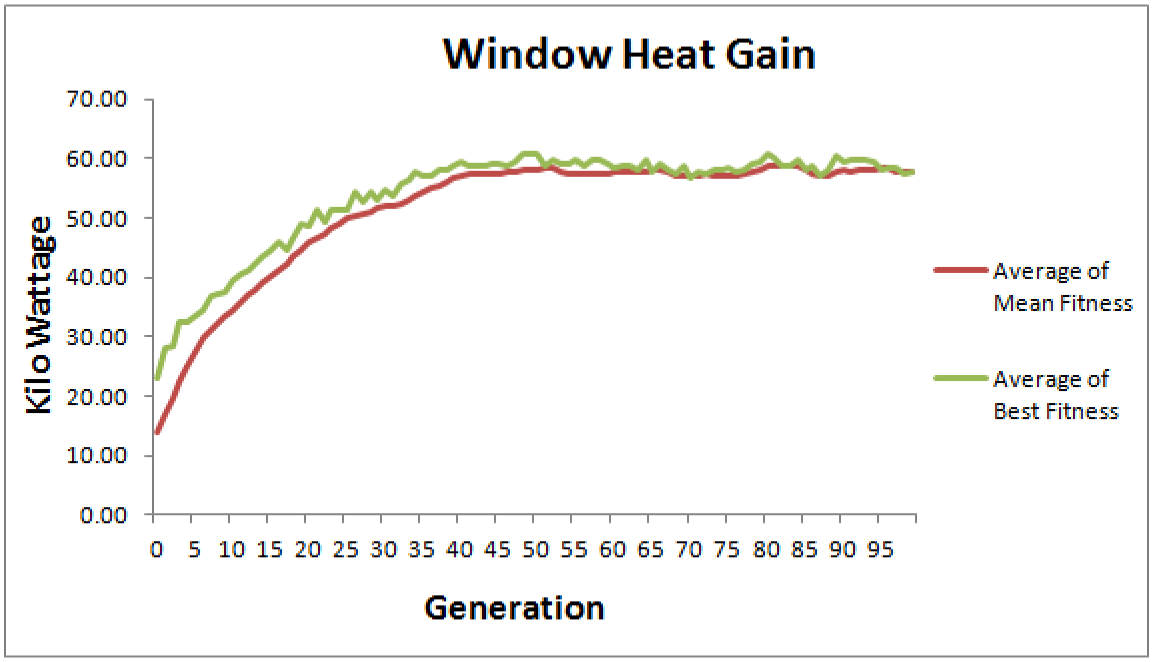


Figure 7.19: Population average window heat gain performance plot.

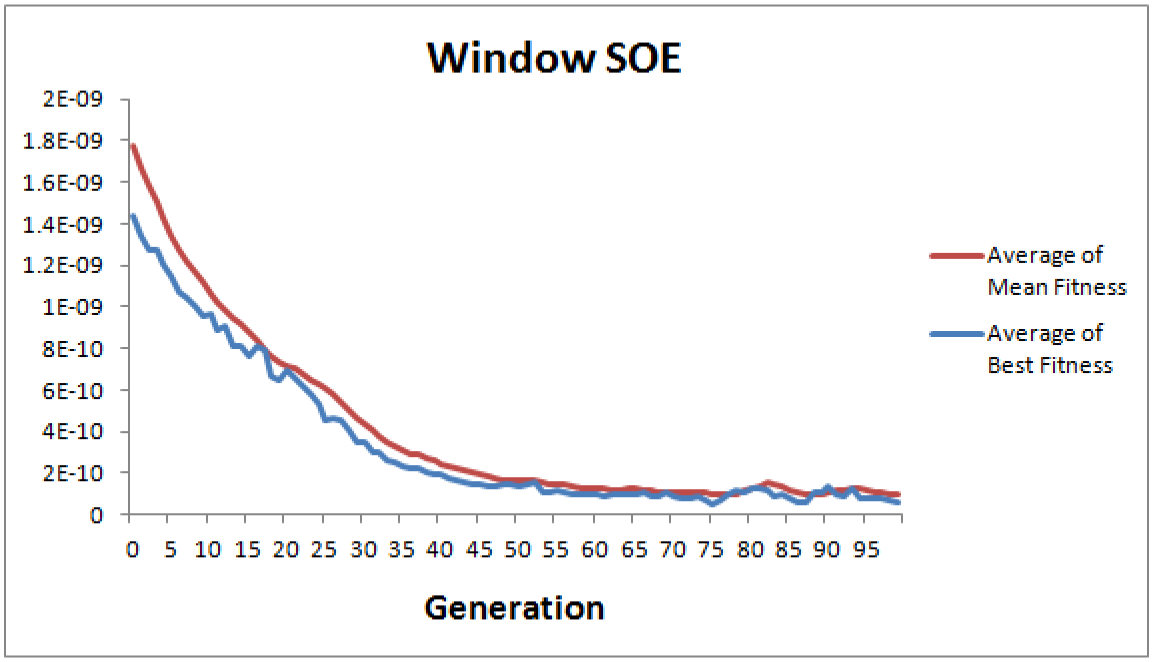


Figure 7.20: Population average window sum of errors.

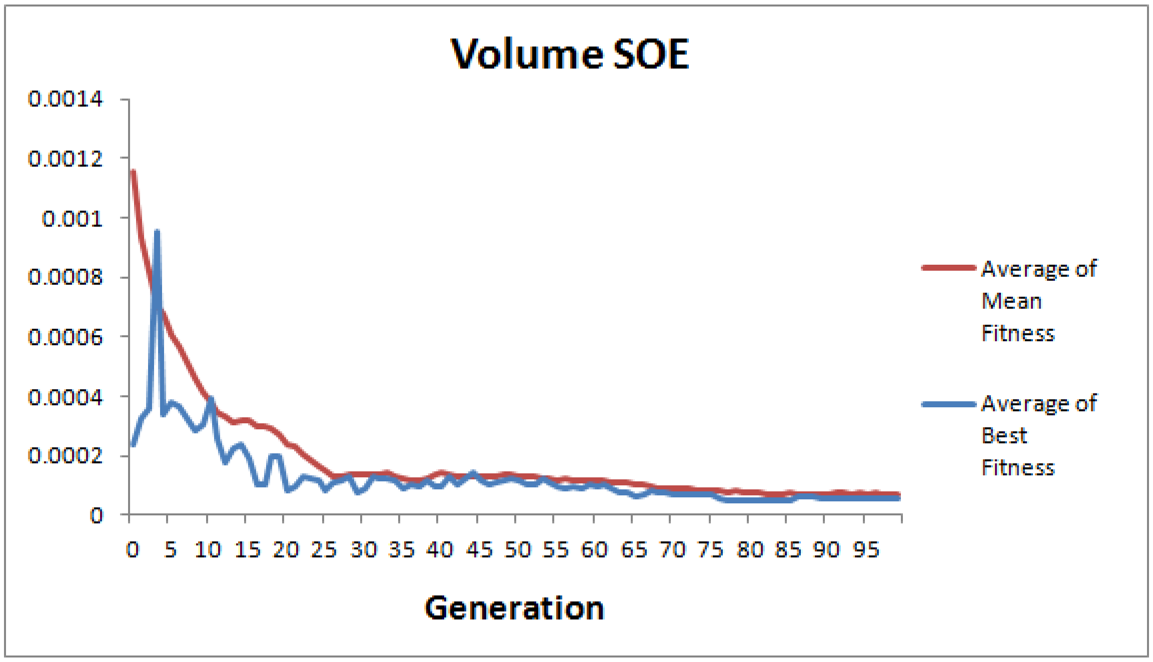


Figure 7.21: Population average volume sum of errors.

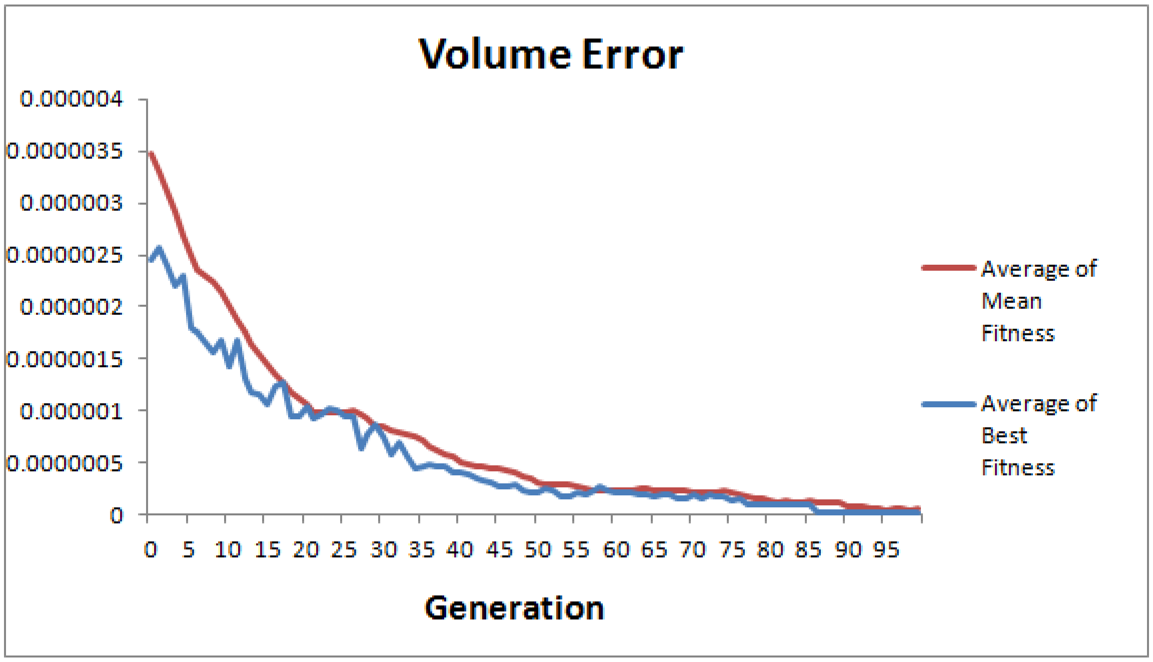
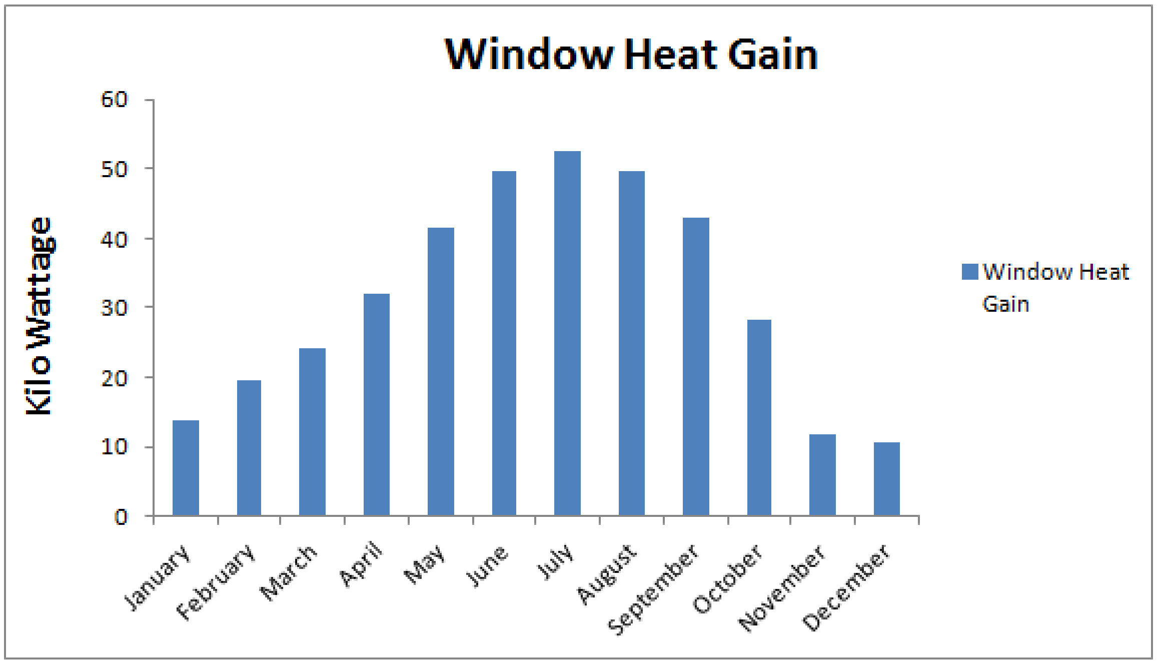
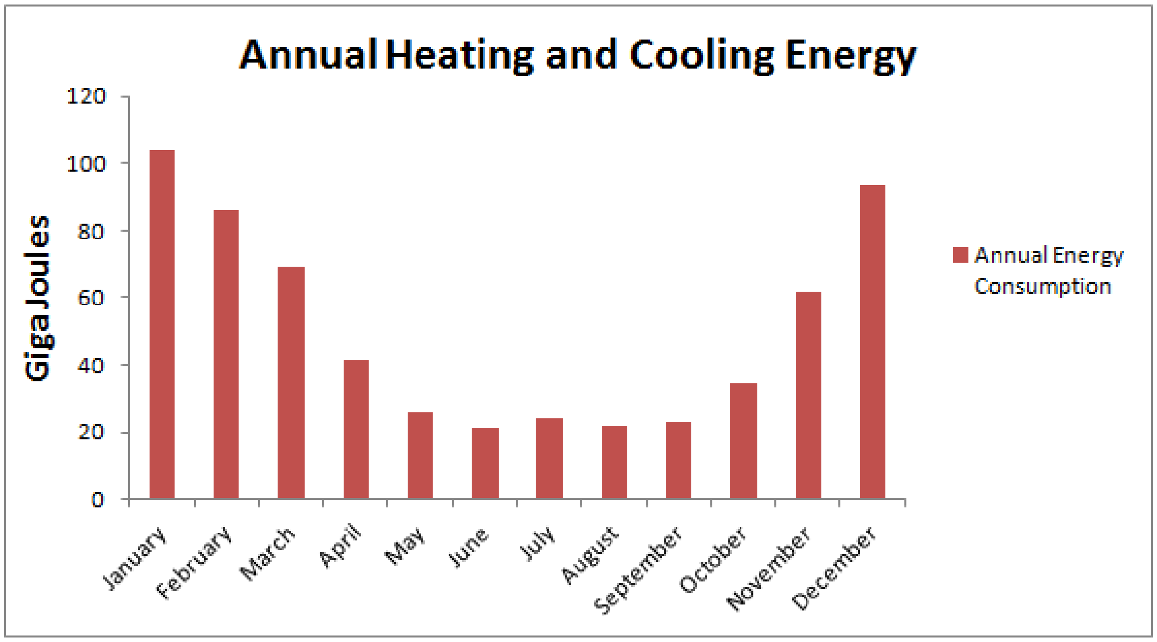


Figure 7.22: Population average volume error.

Figure 7.23 shows annual energy usage and window heat gain of the best model of all runs. Comparing Figure 7.23a with Figure 7.9a and 7.16a upholds our statement about the differences between this experiment, and those in Section 7.3, and 7.4. Window heat gain of the best model of this experiment is more than that in Section 7.3, because of the higher window area facing south of the models of this experiment. With the same reasoning, window heat gain in Section 7.4 is more than the window heat gain here. The comparison of the Figure 7.23b and 7.9b from Section 7.3, determines that the annual energy consumption of the best model is more than the best model of Section 7.3, because the volume of the best model is more than that experiment. Also, by having 35% window area on all walls, the window heat loss is more than in Section 7.4. Therefore, annual energy consumption of here must be more than the annual energy in Section7.4. A comparison of Figure 7.23b and Figure 7.16b, confirms that the annual energy consumption of the best model is more.



(a) Window heat gain



(b) Annual energy consumption

Figure 7.23: Annual energy consumption and window heat gain of the best model of all trials.

Figure 7.24 illustrates the best model of the 10 trials. This model has the window area of 32.25%, 33.50%, 36.37%, and 33.50% on north, east, south, and west wall respectively. Also its volume is 9950 m^3.

Table 7.4 shows material usage of the best 10 solutions. Among all walls, Wall_5 (dense brick, insulation, concrete, and gypsum) is the most preferable wall, since 78% of the walls selected by the best 10 solutions are Wall_5. This wall is the third best wall in U-factor. All the selected windows are double pane and Window_2 (two 3 mm glass with argon in between) is selected 71% of the time by the best 10 solutions. This window has the second best U-factor and the best SHGC among all two layers windows. Roof_1 (no mass with thermal resistance 65%) is the frequent selected roof. This roof has the biggest U-factor among all roofs in our system. The floor with the biggest U-factor, Floor_2 (concrete, and hardwood, is selected 82% of the time by the best 10 solutions. Finally, the glass door is the most frequently selected door. This door is the only glass door our system has.

Construction	**% usage**
Wall_1	7%
Wall_2	15%
Wall_5	78%
Window_1	16%
Window_2	71%
Window_4	13%
Roof_1	68%
Roof_2	5%
Roof_3	26%
Floor_1	17%
Floor_2	82%
Door_1	20%
Door_2	20%
Door_3	60%

Table 7.5: Material analysis of the best 10 solutions.

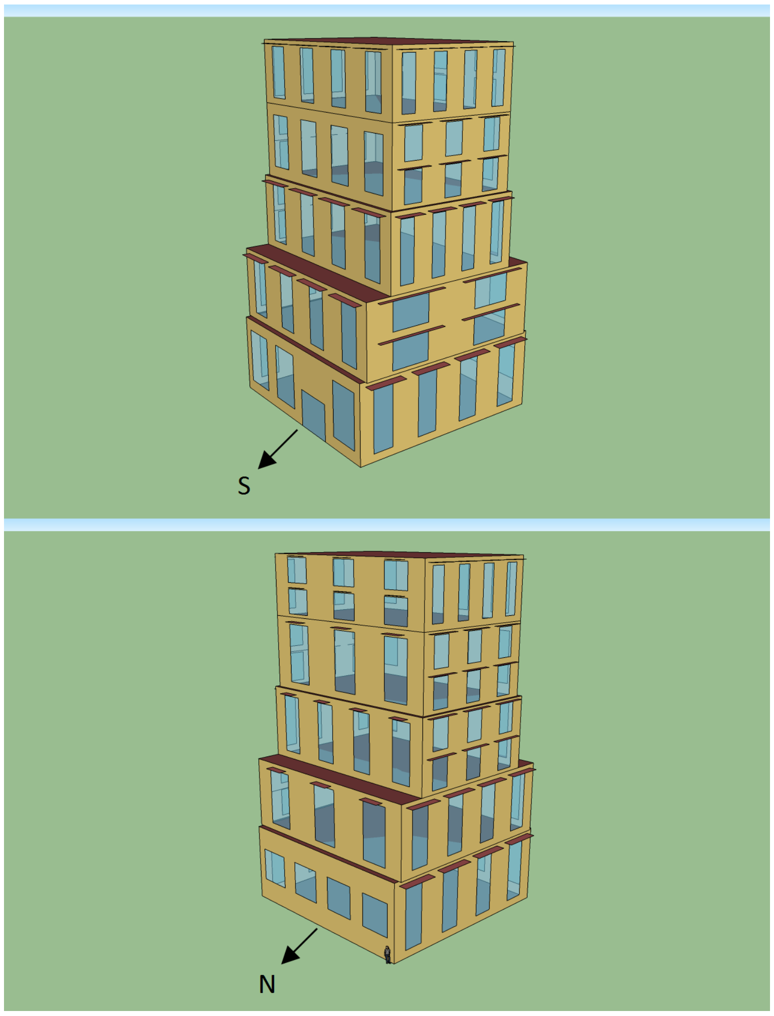


Figure 7.24: The best model of 10 trials.

7.6 Summary

This chapter investigated multi-floors models. For the sake of simplicity in comparing models, all models had five floors. Window heat gain for winter and annual energy consumption were the two energy based objective used in all experiments. Alongside the two energy based objective, two window constraints and two volume constraints were examined as well.

Experiments determined that having at least 25% window area is easier for GP than having exactly 35% window area. The reason is that because of the two energy based objectives (maximizing window heat gain for winter, and minimizing annual energy consumption), GP inclines to make models with larger windows facing south and have no windows on other sides. Therefore, at least 25% window constraint does not stop models from having bigger windows facing south, while the exact 35% window area has to stop models from attaining windows larger than 35%. In addition, the GP propensity to have smaller windows for north, east, and west, is reached easier by having 25% window area constraint than having 35% window area.

The first volume constraint forces GP to produce models that upper floors are smaller and lower floors are larger. The second volume constraint adjusts the total volume of the building. GP shows that the first volume constraint is harder to achieve than the second volume constraint.

Wall_5 (dense brick, insulation, concrete, and gypsum) , Window_2 (3 mm glass, 13 mm argon, and 3 mm glass), Roof_1 (no mass with thermal resistance 65%), Floor_2 (concrete, and hardwood), and Door_3 (single layer 3 mm grey glass door) are the preferred construction used in all experiments of this chapter.

Chapter 8

Discussion

This chapter compares this work with related research. We compare our system with selected related work in evolutionary design and 3D modeling, and evolutionary design and energy efficient architecture.

8.1 Evolutionary Design and 3D Modeling

Hornby used GA and used L-system grammars to design tables and robots[25]. He benefited from modularity in his designs. Later, he defined regularity and hierarchy as other design elements [24]. In contrast, we used GP as our evolutionary system and a split grammar to design models. Modularity and regularity were implicitly embedded in our grammar.

McDermott *et al.* [32] used string-rewriting grammars to evolve architectural designs. In their work, rules were set prior to evolutionary process and then GA selects rules when it is required. We used string-rewriting as well, with the difference that we used GP to select rules and use design elements.

O'Reilly *et al.* [39] used shape grammars a grammatical evolution system to generate models. A similarity between their work and ours is that only the surface of the model is considered, and not the internal design of models. Our split grammar focused on cubic-shape structures. They treat the problem as a single objective one, while we use a multi-objective approach.

O'Neill *et al.* used grammatical evolution and shape grammar to design shelters [38], whereas we used split grammar. They used a semi-interactive fitness evaluation method to evaluate models, while ours is fully automated.

Coia [17] used GP and split grammars to design conceptual building architectures. His results were very complex compared to ours. The results were not realistic to build or analyze with respect to energy performance.

Bergen [8] used GP and L-system to design aesthetically pleasing 3D models. L-systems create complex recursive structures, unlike our split grammar. Our system's models were more practical to evaluate. The drawback of our method is that less complex models are derived. Both systems use normalized rank sum.

8.2 Evolutionary Design and Energy Efficient Architecture

8.2.1 Turrin et al. [46]

Turrin et al.'s goal was to optimize the large roof of a pre-defined structure, while our goal was to make a whole building by defining sizes, materials, and structures. They accepted curved glass roofs as well to identify the best roof structure, while we only accepted planar roofs. In addition, they used a GA as their evolutionary design system which tries to optimize the parameters, while we used GP and split grammars to design optimized models. To evaluate models, designers evaluated models by considering the output of an energy simulation system alongside their opinions about the design's appearance. Our system was completely automated, and aesthetics were not considered.

8.2.2 Malkawi et al. [30]

They used computational fluid dynamics (CFD) to calculate temperature and air velocity contours. The method that EnergyPlus uses is simpler and less accurate but faster than CFD. In their work, a GA is used to identify a window, a door, a supply duct, a return duct placement, and room sizes. Our GP system is capable of generating more than one window and door. Also we considered skylights, different type of roofs, overhangs, different materials, and multiple floors. They penalized the models which do not meet constraints, while we force the grammar to make models which meet constraints. They evaluated models interactively, while we use automated evolution. They considered temperature and ventilation. We also considered energy usage, and window heat gain plus window and volume restrictions. In addition, we considered geographical locations and corresponding weather conditions.

8.2.3 Marin et al. [31]

Two chromosomes are used in their system, one for topology and the other to design facades. Our GP system used one tree to handle both of them. Winter comfort is considered in their system, while we consider thermal comfort for the whole year. Window heat gain, volume and window constraints were other objectives we considered. Their system is capable of making impractical buildings, while ours tend to be more realistic due to a split grammar constraints. The amount of energy needed for heating during winter is the objective used in their experiments, while we also consider energy for cooling, window heat gain, and volume and window constraints.

8.2.4 Harrington [22]

GP and L-systems were used to evolve structures in his work, while in our work GP and split grammar is used. Maximizing sun exposure in winter and minimizing sun exposure in summer was his goal. These goals have implicit impacts on energy consumption for heating and cooling. We used annual energy consumption directly. He does not consider material, facade design, different type of roofs, skylights, and overhangs, which we considered. He considered a group of building to maximize the sum of sun exposures to all buildings of the group. In our research, the impact of other buildings in blocking the sun is not considered. Normalized rank sum is the multi-objective strategy chosen in his and our work.

8.2.5 Caldas [11, 12, 13, 14, 15]

Material, windows, doors, overhangs, and roofs are considered in both hers and our work. Although windows and overhangs are used in both, windows and overhangs were treated more flexibly in our system. Cost of materials and design elements are calculated in her work, while we never consider cost. She used GENE_ARCH which is a GA generative design system. Our GP system in conjunction with split grammars and string rewriting CFG gives more flexibility in designing models. DOE2 is her energy performance and simulation system, while we used EnergyPlus. EnergyPlus uses the best modules of DOE2 and BLAST and puts them together to compute more accurate results in a reasonable time. She used energy consumption and illumination as objectives, while we used energy consumption and window heat gain as objectives. Illumination seems to be more rational at the first glance, but all the standard methods of measuring illumination are based on two selected points in the space of the building. This can add bias to the system while window heat gain does not add bias. Volume is the other objective we used as a separate objective, but because of the limitation of the number of objectives in her work, she combined the volume and energy consumption into one single objective. Pareto ranking with the limitation of at most 2 objectives at a time is used in her work, while in our work we used normalized rank sum. 5 is the most number of objectives we used in experiments, and our system is capable of having more objectives. With Pareto ranking, multiple best solutions are output, while with normalized rank sum one balanced solution with respect to objectives is produced.

8.2.6 Yu [49]

She considered office occupancy to utilize lighting, heating, ventilation, and air conditioning efficiently. We considered annual energy consumption, window heat gain, window constraints and volume constraints to design energy efficient models. Also, we used weather data for different locations, while she used data from different offices rather than different geographical locations. A single objective is used in her system, while we use multi-objective fitness.

8.2.7 Bouchlaghem [9]

We used GP in conjunction with grammars to design buildings while his work was a parameter based system. Therefore, our system is more flexible and broad compared to his work. To analyze the energy usage of a model, we considered annual energy consumption. He used a winter month and a summer month for analysis. In addition, our system was multi-objective, while his system was single objective. His objective was to minimize the degree of discomfort, while we used the occupant discomfort to calculate energy usage of the model. Window shading was considered in both his and our work.

8.2.8 Shea et al. [42]

They considered lighting performance, while we considered energy performance. In both systems, walls and roofs can be divided into smaller pieces. In their system, smaller components can have different materials while in our system windows of a wall have the same material. Both used the multi-objective approach. They used Pareto ranking and we used normalized rank sum.

Chapter 9

Conclusions and Future Work

In this chapter, we provide a brief summary of what we have done in this research and the results we have obtained. This chapter also contains suggestions for how to extend this work through enhancing design elements, materials, and modeling language.

9.1 Conclusion

In this research, we developed an evolutionary design system to create 3D building models considering passive solar energy. GP is the evolutionary system used in this research. A split grammar is used to create geometries, build models, and select materials. EnergyPlus is the energy simulation and analysis system used. Multiple constraints were treated in a multi-objective fashion, using normalized rank sum. Different experiments used different combination of the following objectives:

- Window heat gain for winter.
- Energy consumption for heating in winter.
- Energy consumption for cooling in summer.
- Annual energy consumption for heating and cooling.
- Window constraints.
- Volume constraints.

Also, there are other items which were used in different experiments:

- Skylights, different kinds of roofs, overhangs.

- Single floor and Multi-floor designs.
- Different geographical locations.

Experimental results show that consistent solutions with respect to size, geometry, and design elements are achieved in all experiments. Considering experimental results, it is realized that specific materials were selected consistently by the best models of each experiment. Also, experimental results show that best models appearance and other design aspects were consistent for each experiment within the design framework permitted by the split grammar. The split grammar constants that we encoded into our grammar such as cubic models, and window layouts add bias to models. In addition, a combination of objectives such as window heat gain, annual energy usage, and window and volume constraints affects the output models evolved by the system. This can affect their energy efficiency as well. Therefore, solutions are not always feasible or fully optimized in terms of energy usage.

In addition, in an experiment we examine the impact of geographical location on models and materials. Regardless of the location, GP chooses the same high-efficiency materials for constructions and facades. Best models of the experiments with the same geographical location were similar to each other.

Finally, one issue during our experiments was their running time. The shortest and the longest experiment we have done used 19 hours (approximately 6 seconds each model) and 550 hours (approximately 50 seconds each model of Section 7.5) CPU time respectively. As with all other evolutionary techniques, fitness evaluation took the most time of the runs. To expedite the fitness evaluation process, we ran multiple copies of EnergyPlus in parallel by means of multi-thread techniques. Still the longest experiment took almost 5 days and 17 hours by making 4 copies of EnergyPlus running simultaneously on a 4 core processor.

9.2 Future work

Expanding the library of materials for walls, windows, roofs, floors, and doors is one way to extend this work. In addition, in this thesis we did not consider the cost of materials in material selection. Minimizing cost of material can be another objective to consider.

A more complex design grammar can be used. We should keep in mind that some enhancements may affect energy simulations and some may not. Also, models with different geometry such as L-shape models, hexagonal models, pyramids, etc. can be considered. More types of roofs, slanted walls, and overhangs with slope are other ways to have more varieties of model geometries.

Other EnergyPlus energy simulation objectives such as, humidity control, window heat loss, pressure control, daylight calculations, transmitted or diffused solar radiant, more complicated window constraints, and volume constraints can be beneficial. Dividing heat gain into radiant, visible, convective, and latent heat gain might be helpful. Also environmental objectives lead models toward more diverse models. For example, although gabled roofs are not energy efficient compare to flat roofs, they are preferred in rainy and/or snowy regions.

The bottleneck in our system is EnergyPlus simulation time. Finding a method to simulate GP individuals in parallel will reduce evolution time. One of the methods is to run energy simulation on GPUs. Although GPU cores are slower than CPU cores, GPUs have more cores and they can simulate the whole population in parallel [41].

In our system, we did not consider aesthetic aspects of models. Aesthetic based objectives can lead models toward more visually pleasing models. One way is to use an interactive evolutionary system. Analyzing all models by a human being is a tedious job, but after every few generations e.g. 10 generations, models can be shown to a human being and he decides which models are preferred in different aspects.

Bibliography

[1] U.S. Environmental Protection Agency. Green building basic information. `http://www.epa.gov/greenbuilding/pubs/about.htm`, October 2010.

[2] Bruce Anderson, Malcolm Wells, Malcolm Wells, and Malcolm Wells. *Passive solar energy: the homeowner's guide to natural heating and cooling.* Brick House Publishing Company, 1981.

[3] W Annicchiarico and M Cerrolaza. Structural shape optimization 3d finite-element models based on genetic algorithms and geometric modeling. *Finite Elements in Analysis and Design*, 37(5):403–415, 2001.

[4] Wolfgang Banzhaf, Frank D. Francone, Robert E. Keller, and Peter Nordin. *Genetic programming: an introduction: on the automatic evolution of computer programs and its applications.* Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 1998.

[5] Peter Bentley. *Evolutionary design by computers.* Morgan Kaufmann, 1999.

[6] Peter J Bentley and Jonathan P Wakefield. Finding acceptable solutions in the pareto-optimal range using multiobjective genetic algorithms. In *Soft Computing in Engineering Design and Manufacturing*, pages 231–240. Springer, 1998.

[7] Peter John Bentley and David W Corne. *Creative evolutionary systems.* Morgan Kaufmann, 2002.

[8] Steve Bergen. Automatic structure generation using genetic programming and fractal geometry. Master's thesis, Department of Computer Science, Brock University, 2011.

[9] N Bouchlaghem. Optimising the design of building envelopes for thermal performance. *Automation in Construction*, 10(1):101–112, 2000.

[10] Jonathan Byrne, Michael Fenton, Erik Hemberg, James McDermott, Michael ONeill, Elizabeth Shotton, and Ciaran Nally. Combining structural analysis and multi-objective criteria for evolutionary architectural design. In *Applications of Evolutionary Computation*, volume 6625 of *Lecture Notes in Computer Science*, pages 204–213. Springer Berlin Heidelberg, 2011.

[11] Luisa Caldas. Generation of energy-efficient architecture solutions applying gene_arch: An evolution-based generative design system. *Advanced Engineering Informatics*, 22(1):59–70, 2008.

[12] Luisa Caldas, Leslie Norford, and João Rocha. An evolutionary model for sustainable design. *Management of Environmental Quality: An International Journal*, 14(3):383–397, 2003.

[13] Luisa G Caldas. Three-dimensional shape generation of low-energy architectural solutions using pareto genetic algorithms. In *Proceedings of the eCAADe Conference*, pages 647–654, 2005.

[14] Luisa G Caldas and Leslie K Norford. Shape generation using pareto genetic algorithms: integrating conflicting design objectives in low-energy architecture. *International journal of architectural computing*, 1(4):503–515, 2003.

[15] Luisa Gama Caldas and Leslie K Norford. A design optimization tool based on a genetic algorithm. *Automation in construction*, 11(2):173–184, 2002.

[16] Carlos A Coello Coello, Gary B Lamont, and David A Van Veldhuisen. *Evolutionary algorithms for solving multi-objective problems.* Springer, 2007.

[17] C. Coia and B.J. Ross. Automatic evolution of conceptual building architectures. In *Evolutionary Computation (CEC), 2011 IEEE Congress on*, pages 1140–1147, 2011.

[18] D. Corne and J. Knowles. Techniques for highly multiobjective optimisation: Some nondominated points are better than others. In *Proceedings GECCO 2007*, pages 773–780. ACM Press, 2007.

[19] Robert W.J. Flack and Brian J. Ross. Evolution of architectural floor plans. In *Applications of Evolutionary Computation*, volume 6625 of *Lecture Notes in Computer Science*, pages 313–322. Springer Berlin Heidelberg, 2011.

[20] Carlos M Fonseca and Peter J Fleming. An overview of evolutionary algorithms in multiobjective optimization. *Evolutionary computation*, 3(1):1–16, 1995.

[21] D.E. Goldberg. *Genetic Algorithms in Search, Optimization, and Machine Learning*. Addison Wesley, 1989.

[22] Adrian Harrington. Enabling and measuring complexity in evolving designs using generative representations for artificial architecture. Master's thesis, Department of Computer Science, Brock University, 2012.

[23] James J. Hirsch. The home of doe-2 based building energy use and cost analysis software. `http://doe2.com/`, July 2013.

[24] Gregory S. Hornby. Measuring, enabling and comparing modularity, regularity and hierarchy in evolutionary design. In *Proceedings of the 2005 conference on Genetic and evolutionary computation*, GECCO 2005, pages 1729–1736, New York, NY, USA, 2005. ACM.

[25] Gregory S Hornby. Generative representations for computer-automated design systems. Technical report, NASA Ames Research Center, pages 269-3, 2004. Retrieved from http://ti.arc.nasa.gov/publications/.

[26] Yan Ji and Stellios Plainiotis. *Design for sustainability*. China Architecture & Building Press, Beijing, 2006.

[27] John R. Koza. *Genetic Programming: On the Programming of Computers by Means of Natural Selection*. MIT Press, Cambridge, MA, USA, 1992.

[28] John R Koza. *Genetic programming II: automatic discovery of reusable programs*. MIT press, 1994.

[29] Derek Scott Linden. *Automated design and optimization of wire antennas using genetic algorithms*. PhD thesis, 1997.

[30] Ali M Malkawi, Ravi S Srinivasan, Yun K Yi, and Ruchi Choudhary. Decision support and design evolution: integrating genetic algorithms, cfd and visualization. *Automation in construction*, 14(1):33–44, 2005.

[31] Philippe Marin, Jean-Claude Bignon, and Hervé Lequay. Generative exploration of architectural envelope responding to solar passive qualities. Retrieved from http://halshs.archives-ouvertes.fr/halshs-00348544, May 2008.

[32] James McDermott, John Mark Swafford, Martin Hemberg, Jonathan Byrne, Erik Hemberg, Michael Fenton, Ciaran McNally, Elizabeth Shotton, and Michael ONeill. String-rewriting grammars for evolutionary architectural design. *Environment and Planning-Part B*, 39(4):713, 2012.

[33] Bridgette Meinhold. A-lab completes stacked energy-efficient statoil headquarters in norway. `http://www.statoil.com/en/Pages/default.aspx`, April 2013.

[34] David J Montana. Strongly typed genetic programming. *Evolutionary computation*, 3(2):199–230, 1995.

[35] U.S. Department of Energy. Building energy software tools directory. `http://apps1.eere.energy.gov/buildings/tools_directory/subjects.cfm/pagename=subjects/pagename_menu=whole_building_analysis/pagename_submenu=energy_simulation`, October 2012.

[36] U.S. Department of Energy. Energyplus energy simulation software. `http://apps1.eere.energy.gov/buildings/energyplus/`, October 2012.

[37] U.S. Department of Energy. Energyplus energy simulation software, weather data. `http://apps1.eere.energy.gov/buildings/energyplus/cfm/weather_data3.cfm/region=4_north_and_central_america_wmo_region_4/country=3_canada/cname=CANADA`, October 2012.

[38] Michael O'Neill, James McDermott, John Mark Swafford, Jonathan Byrne, Erik Hemberg, Anthony Brabazon, Elizabeth Shotton, Ciaran McNally, and Martin Hemberg. Evolutionary design using grammatical evolution and shape grammars: Designing a shelter. *International Journal of Design Engineering*, 3(1):4–24, 2010.

[39] Una-May OReilly and Martin Hemberg. Integrating generative growth and evolutionary computation for form exploration. *Genetic Programming and Evolvable Machines*, 8(2):163–186, 2007.

[40] Riccardo Poli, William B. Langdon, and Nicholas Freitag McPhee. *A field guide to genetic programming*. Published via `http://lulu.com`, 2008.

[41] Jason Sanders and Edward Kandrot. *CUDA by example: an introduction to general-purpose GPU programming*. Addison-Wesley Professional, 2010.

[42] Kristina Shea, Andrew Sedgwick, and Giulio Antonuntto. Multicriteria optimization of paneled building envelopes using ant colony optimization. In *Intelligent Computing in Engineering and Architecture*, pages 627–636. Springer, 2006.

[43] George Stiny. *Shape: talking about seeing and doing*. The MIT Press, 2008.

[44] George Stiny and James Gips. Shape grammars and the generative specification of painting and sculpture. *Information processing*, 71(1460-1465), 1972.

[45] Trimble. Sketchup to create, modify and share your 3d models. `http://www.sketchup.com/intl/en/download/gsu.html`, October 2012.

[46] Michela Turrin, Peter von Buelow, Rudi Stouffs, and Axel Kilian. Performance-oriented design of large passive solar roofs. In *Future Cities: ECAADE 2010: Proceedings of the 28th Conference on Education in Computer Aided Architectural Design in Europe, September 15-18, 2010, Zurich, Switzerland, ETH Zurich*, page 321. vdf Hochschulverlag AG, 2010.

[47] Peter Wonka, Michael Wimmer, François Sillion, and William Ribarsky. Instant architecture. *ACM Trans. Graph.*, 22(3):669–677, July 2003.

[48] Jonathan A Wright, Heather A Loosemore, and Raziyeh Farmani. Optimization of building thermal design and control by multi-criterion genetic algorithm. *Energy and Buildings*, 34(9):959–972, 2002.

[49] T. Yu. Modeling occupancy behavior for energy efficiency and occupants comfort management in intelligent buildings. In *Machine Learning and Applications (ICMLA), 2010 Ninth International Conference on*, pages 726–731. IEEE, 2010.

Appendix A

List of input to EnergyPlus

Table A.1 consists information about material used in our system. Ro, Th, Co, De, SH, TA, SA, and VA are roughness, thickness, conductivity, density, specific heat, thermal absorptance, solar absorptance,and visible absorptance respectively.

Name	**Ro**	**Th**	**Co**	**De**	**SH**	**TA**	**SA**	**VA**
RG01	Rough	0.012	1.44	881	1674	.9	.65	.65
BR01	VeryRough	.009	.162	1121	1464	.9	.7	.7
IN46	VeryRough	0.076	.23	24	1590	.9	.5	.5
WD01	MediumSmooth	0.019	.115	513	1381	.9	.78	.78
PW03	MediumSmooth	.012	.115	545	1213	.9	.78	.78
IN02	Rough	.09	.04	10	837	.9	.75	.75
GP01	MediumSmooth	.012	.16	801	837	.9	.75	.75
GP02	MediumSmooth	0.015	.16	801	837	.9	.75	.75
CC03	MediumRough	.101	1.31	2243	837	.9	.65	.65
PB01	MediumSmooth	0.12	.16	950	840	.9	.6	.6
FQ01	Rough	.066	.04	12	840	.9	.6	.6
WS01	Rough	.009	.14	530	900	.9	.6	.6
PB02	Rough	.010	.16	950	840	.9	.6	.6
FQ02	Rough	.111	.04	12	840	.9	.6	.6
RD01	Rough	.019	.14	530	900	.9	.6	.6
HFC5	MediumRough	.101	1.7	2243	837	.9	.65	.65

Table A.1: Material

Name	**Roughness**	**Thermal Resistance**	**TA**	**SA**	**VA**
MatClng01	Rough	0.652	.65	.65	.65

Table A.2: Material No Mass

Name	**A1**	**A2**	**A3**	**A4**	**A5**	**A6**	**A7**	**A8**	**A9**	**A10**	**A11**	**A12**
CL3M	SA	.003	.83	.07	.07	.90	.08	.08	0	.84	.84	.9
GR3M	SA	.003	.63	.06	.06	.61	.06	.06	0	.84	.84	.9

Table A.3: Window Material Glazing

Name	**Gas Type**	**Thickness**
A13M	Air	.0127

Table A.4: Window Material Gas

Material for windows are different and they are categorized into two categories: glazing, and gas.

Data needed for glazing window are optical data type, thickness, solar transmittance at normal incidence, front side solar reflectance at normal incidence, back side solar reflectance at normal incidence, visible transmittance at normal incidence, front side visible reflectance at normal incidence, back side visible reflectance at normal incidence, infrared transmittance at normal incidence, front side infrared hemispherical emissivity, back side infrared hemispherical emissivity, conductivity, and dirt correction factor for solar and visible transmittance.

Data which are needed to specify a gas window material are gas type, and thickness. Table A.3 and Table A.4 consists of all glazing window material and gas window used in our system.

For the sake of simplicity, ideal loads air system is used as a cooling and heating device. This air system has an infinite power and can change temperature to any degree.

As we mentioned in section 4.6 EnergyPlus weather file contains static data about the location of the building on the earth. These data encompass latitude, longitude, time zone, elevation of the location, and ground temperature. Also hourly information of dry bulb temperature, dew point temperature, relative humidity, atmospheric pressure, extraterrestrial horizontal radiation, extraterrestrial direct normal radiation, horizontal infrared radiation intensity from sky, global horizontal radiation, direct normal radiation, diffuse horizontal radiation, global horizontal illuminance, direct normal illuminance, diffuse horizontal illuminance, zenith luminance, wind direction, wind speed, total sky cover, opaque sky cover, visibility, ceiling height, present weather observation, precipitable water, aerosol optical depth, snow depth, days since last snow, albedo, rain, and rain quantity are provided in EPW files.

Appendix B

List of output from EnergyPlus

Zone,Average,Outdoor Dry Bulb (C)

Zone,Average,Outdoor Dew Point (C)

Zone,Average,Outdoor Wet Bulb (C)

Zone,Average,Outdoor Humidity Ratio (kgWater/kgDryAir)

Zone,Average,Outdoor Relative Humidity (%)

Zone,Average,Outdoor Barometric Pressure (Pa)

Zone,Average,Wind Speed (m/s)

Zone,Average,Wind Direction (deg)

Zone,Average,Sky Temperature (C)

Zone,Average,Horizontal Infrared Radiation Intensity (W/m2)

Zone,Average,Diffuse Solar (W/m2)

Zone,Average,Direct Solar (W/m2)

Zone,Average,Liquid Precipitation (mm)

Zone,Average,Ground Reflected Solar (W/m2)

Zone,Average,Ground Temperature (C)

Zone,Average,Surface Ground Temperature (C)

Zone,Average,Deep Ground Temperature (C)

Zone,Average,FCFactor Ground Temperature (C)

Zone,Average,Outdoor Enthalpy (J/kg)

Zone,Average,Outdoor Air Density (kg/m3)

Zone,Average,Solar Azimuth Angle (deg)

Zone,Average,Solar Altitude Angle (deg)

Zone,Average,Solar Hour Angle (deg)

Zone,Average,Fraction of Time Raining ()

Zone,Average,Fraction of Time Snow On Ground ()

Zone,Average,Exterior Horizontal Illuminance From Sky (lux)

Zone,Average,Exterior Horizontal Beam Illuminance (lux)

Zone,Average,Exterior Beam Normal Illuminance (lux)

Zone,Average,Luminous Efficacy of Sky Diffuse Solar Radiation (lum/W)

Zone,Average,Luminous Efficacy of Beam Solar Radiation (lum/W)

Zone,Average,Sky Clearness for Daylighting Calculation ()

Zone,Average,Sky Brightness for Daylighting Calculation ()

Zone,Average,Daylight Saving Time Indicator ()

Zone,Average,DayType Index ()

Zone,Average,Water Mains Temperature (C)

Zone,Average,Zone Outdoor Dry Bulb (C)

Zone,Average,Zone Outdoor Wet Bulb (C)

Zone,Average,Zone Outdoor Wind Speed (m/s)

Zone,Sum,Zone Total Internal Radiant Heat Gain (J)

Zone,Average,Zone Total Internal Radiant Heat Gain Rate (W)

Zone,Sum,Zone Total Internal Visible Heat Gain (J)

Zone,Average,Zone Total Internal Visible Heat Gain Rate (W)

Zone,Sum,Zone Total Internal Convective Heat Gain (J)

Zone,Average,Zone Total Internal Convective Heat Gain Rate (W)

Zone,Sum,Zone Total Internal Latent Gain (J)

Zone,Average,Zone Total Internal Latent Gain Rate (W)

Zone,Sum,Zone Total Internal Total Heat Gain (J)

Zone,Average,Zone Total Internal Total Heat Gain Rate (W)

Zone,Average,Zone Transmitted Solar (W)

Zone,Average,Zone Beam Solar from Exterior Windows (W)

Zone,Average,Zone Beam Solar from Interior Windows (W)

Zone,Average,Zone Diff Solar from Exterior Windows (W)

Zone,Average,Zone Diff Solar from Interior Windows (W)

Zone,Average,Zone Window Heat Gain (W)

Zone,Average,Zone Window Heat Loss (W)

Zone,Sum,Zone Transmitted Solar Energy (J)

Zone,Sum,Zone Beam Solar from Exterior Windows Energy (J)

Zone,Sum,Zone Beam Solar from Interior Windows Energy (J)

Zone,Sum,Zone Diff Solar from Exterior Windows Energy (J)

Zone,Sum,Zone Diff Solar from Interior Windows Energy (J)

Zone,Sum,Zone Window Heat Gain Energy (J)

Zone,Sum,Zone Window Heat Loss Energy (J)
Zone,Average,Surface Ext Sunlit Area (m2)
Zone,Average,Surface Ext Sunlit Fraction ()
Zone,Average,Surface Ext Solar Incident (W/m2)
Zone,Average,Surface Ext Solar Beam Incident (W/m2)
Zone,Average,Surface Ext Solar Sky Diffuse Incident (W/m2)
Zone,Average,Surface Ext Solar Ground Diffuse Incident (W/m2)
Zone,Average,Surface Ext Solar Beam Cosine Of Incidence Angle ()
Zone,Average,Surface Ext Solar From Sky Diffuse Refl From Ground (W/m2)
Zone,Average,Surface Ext Solar From Sky Diffuse Refl From Obstructions (W/m2)
Zone,Average,Surface Ext Beam Sol From Bm-To-Bm Refl From Obstructions (W/m2)
Zone,Average,Surface Ext Diff Sol From Bm-To-Diff Refl From Obstructions (W/m2)
Zone,Average,Surface Ext Solar From Bm-To-Diff Refl From Ground (W/m2)
Zone,Average,Surface Anisotropic Sky Multiplier ()
Zone,Average,Window Solar Absorbed:All Glass Layers (W)
Zone,Average,Total Shortwave Absorbed:All Glass Layers (W)
Zone,Average,Window Transmitted Solar (W)
Zone,Average,Window Transmitted Beam Solar (W)
Zone,Average,Window Transmitted Beam-to-Beam Solar (W)
Zone,Average,Window Transmitted Beam-to-Diffuse Solar (W)
Zone,Average,Window Transmitted Diffuse Solar (W)
Zone,Average,Window Heat Gain (W)
Zone,Average,Window Heat Loss (W)
Zone,Average,Window Gap Convective Heat Flow (W)
Zone,Average,Window Solar Absorbed:Shading Device (W)
Zone,Sum,Window Solar Absorbed:All Glass Layers Energy (J)
Zone,Sum,Window Transmitted Solar Energy (J)
Zone,Sum,Window Transmitted Beam Solar Energy (J)
Zone,Sum,Window Transmitted Beam-to-Beam Solar Energy (J)
Zone,Sum,Window Transmitted Beam-to-Diffuse Solar Energy (J)
Zone,Sum,Window Transmitted Diffuse Solar Energy (J)
Zone,Sum,Window Heat Gain Energy (J)
Zone,Sum,Window Heat Loss Energy (J)
Zone,Sum,Window Gap Convective Heat Flow Energy (J)
Zone,Sum,Window Solar Absorbed:Shading Device Energy (J)
Zone,Average,Window System Solar Transmittance ()

Zone,Average,Window System Solar Reflectance ()

Zone,Average,Window System Solar Absorptance ()

Zone,Average,Inside Glass Condensation Flag ()

Zone,Average,Inside Frame Condensation Flag ()

Zone,Average,Inside Divider Condensation Flag ()

Zone,Average,Beam Solar Reflected by Outside Reveal Surfaces (W)

Zone,Sum,Beam Solar Reflected by Outside Reveal Surfaces Energy (J)

Zone,Average,Solar Horizontal Profile Angle (deg)

Zone,Average,Solar Vertical Profile Angle (deg)

Zone,Average,Glass Beam-Beam Solar Transmittance ()

Zone,Average,Glass Beam-Diffuse Solar Transmittance ()

Zone,Average,Glass Diffuse-Diffuse Solar Transmittance ()

Zone,Average,Window Calculation Iterations ()

Zone,Average,Beam Sol Intensity from Ext Windows on Inside of Surface (W/m2)

Zone,Average,Beam Sol Amount from Ext Windows on Inside of Surface (W)

Zone,Average,Beam Sol Intensity from Int Windows on Inside of Surface (W/m2)

Zone,Average,Beam Sol Amount from Int Windows on Inside of Surface (W)

Zone,Average,Initial Transmitted Diffuse Solar Absorbed on Inside of Surface (W)

Zone,Average,Initial Transmitted Diffuse Solar Transmitted Out Through Inside of Window Surface (W)

Zone,Average,Total Shortwave Radiation Absorbed on Inside of Surface (W)

Zone,Sum,Beam Sol Amount from Ext Windows on Inside of Surface Energy (J)

Zone,Sum,Beam Sol Amount from Int Windows on Inside of Surface Energy (J)

Zone,Average,debug DifShdgRatioIsoSky ()

Zone,Average,debug DifShdgRatioHoriz ()

Zone,Average,debug WithShdgIsoSky ()

Zone,Average,debug WoShdgIsoSky ()

Zone,Average,Surface Inside Face Temperature (C)

Zone,Average,Surface Outside Face Temperature (C)

Zone,Average,Surface Inside Face Adjacent Air Temperature (C)

Zone,Average,Surface Inside Face Convection Heat Transfer Coefficient (W/m2-K)

Zone,Average,Surface Inside Face Convection Heat Gain Rate (W)

Zone,Average,Surface Inside Face Convection Heat Gain Rate per Area (W/m2)

Zone,Sum,Surface Inside Face Convection Heat Gain Energy (J)

Zone,Average,Surface Inside Face Net Surface Thermal Radiation Heat Gain Rate (W)

Zone,Average,Surface Inside Face Net Surface Thermal Radiation Heat Gain Rate per Area (W/m2)
Zone,Sum,Surface Inside Face Net Surface Thermal Radiation Heat Gain Energy (J)
Zone,Average,Surface Inside Face Solar Radiation Heat Gain Rate (W)
Zone,Average,Surface Inside Face Solar Radiation Heat Gain Rate per Area (W/m2)
Zone,Sum,Surface Inside Face Solar Radiation Heat Gain Energy (J)
Zone,Average,Surface Inside Face Lights Radiation Heat Gain Rate (W)
Zone,Average,Surface Inside Face Lights Radiation Heat Gain Rate per Area (W/m2)
Zone,Sum,Surface Inside Face Lights Radiation Heat Gain Energy (J)
Zone,Average,Surface Inside Face Internal Gains Radiation Heat Gain Rate (W)
Zone,Average,Surface Inside Face Internal Gains Radiation Heat Gain Rate per Area (W/m2)
Zone,Sum,Surface Inside Face Internal Gains Radiation Heat Gain Energy (J)
Zone,Average,Surface Inside Face System Radiation Heat Gain Rate (W)
Zone,Average,Surface Inside Face System Radiation Heat Gain Rate per Area (W/m2)
Zone,Sum,Surface Inside Face System Radiation Heat Gain Energy (J)
Zone,Average,Surface Outside Face Outdoor Air Dry Bulb Temperature (C)
Zone,Average,Surface Outside Face Outdoor Air Wet Bulb Temperature (C)
Zone,Average,Surface Outside Face Outdoor Wind Velocity (m/s)
Zone,Average,Surface Outside Face Convection Heat Gain Rate (W)
Zone,Average,Surface Outside Face Convection Heat Gain Rate per Area (W/m2)
Zone,Sum,Surface Outside Face Convection Heat Gain Energy (J)
Zone,Average,Surface Outside Face Convection Heat Transfer Coefficient (W/m2-K)
Zone,Average,Surface Outside Face Net Thermal Radiation Heat Gain Rate (W)
Zone,Average,Surface Outside Face Net Thermal Radiation Heat Gain Rate per Area (W/m2)
Zone,Sum,Surface Outside Face Net Thermal Radiation Heat Gain Energy (J)
Zone,Average,Surface Outside Face Thermal Radiation to Air Heat Transfer Coefficient (W/m2-K)
Zone,Average,Surface Outside Face Thermal Radiation to Sky Heat Transfer Coefficient (W/m2-K)
Zone,Average,Surface Outside Face Thermal Radiation to Ground Heat Transfer Coefficient (W/m2-K)
Zone,Average,Surface Outside Face Solar Radiation Heat Gain Rate (W)
Zone,Average,Surface Outside Face Solar Radiation Heat Gain Rate per Area (W/m2)
Zone,Sum,Surface Outside Face Solar Radiation Heat Gain Energy (J)

Zone,Average,Opaque Surface Inside Face Beam Solar Absorbed (W)

Zone,Average,Fraction of Time Shading Device Is On ()

Zone,Average,Storm Window On/Off Flag ()

Zone,Average,Window Blind Slat Angle (deg)

Zone,Average,Zone Mean Radiant Temperature (C)

Zone,Average,Zone Mean Air Temperature (C)

Zone,Average,Zone Operative Temperature (C)

Zone,Average,Zone Mean Air Dewpoint Temperature (C)

Zone,Average,Zone Mean Air Humidity Ratio (kgWater/kgDryAir)

HVAC,Average,Zone Air Balance Internal Convective Gains Rate (W)

HVAC,Average,Zone Air Balance Surface Convection Rate (W)

HVAC,Average,Zone Air Balance Interzone Air Transfer Rate (W)

HVAC,Average,Zone Air Balance Outdoor Air Transfer Rate (W)

HVAC,Average,Zone Air Balance System Air Transfer Rate (W)

HVAC,Average,Zone Air Balance Air Energy Storage Rate (W)

HVAC,Sum,Zone/Sys Sensible Heating Energy (J)

HVAC,Sum,Zone/Sys Sensible Cooling Energy (J)

HVAC,Average,Zone/Sys Sensible Heating Rate (W)

HVAC,Average,Zone/Sys Sensible Cooling Rate (W)

HVAC,Average,Zone/Sys Air Temperature (C)

HVAC,Average,Zone/Sys Air Temperature at Thermostat (C)

HVAC,Average,Zone Air Humidity Ratio ()

HVAC,Average,Zone Air Relative Humidity (%)

HVAC,Average,Zone/Sys Sensible Load Predicted (W)

HVAC,Average,Zone/Sys Sensible Load to Heating Setpoint Predicted (W)

HVAC,Average,Zone/Sys Sensible Load to Cooling Setpoint Predicted (W)

HVAC,Average,Zone/Sys Moisture Load Rate Predicted (kgWater/s)

HVAC,Average,Zone/Sys Moisture Load Rate Predicted to humidifying setpoint (kg-Water/s)

HVAC,Average,Zone/Sys Moisture Load Rate Predicted to dehumidifying setpoint (kgWater/s)

Zone,Average,Zone/Sys Thermostat Control Type ()

Zone,Average,Zone/Sys Thermostat Heating Setpoint (C)

Zone,Average,Zone/Sys Thermostat Cooling Setpoint (C)

HVAC,Sum,HVACManage Iterations ()

HVAC,Sum,AirLoop-Zone Iterations ()

HVAC,Sum,Ideal Loads Sensible Heating Energy (J)
HVAC,Sum,Ideal Loads Latent Heating Energy (J)
HVAC,Sum,Ideal Loads Total Heating Energy (J)
HVAC,Sum,Ideal Loads Sensible Cooling Energy (J)
HVAC,Sum,Ideal Loads Latent Cooling Energy (J)
HVAC,Sum,Ideal Loads Total Cooling Energy (J)
HVAC,Sum,Ideal Loads Zone Sensible Heating Energy (J)
HVAC,Sum,Ideal Loads Zone Latent Heating Energy (J)
HVAC,Sum,Ideal Loads Zone Total Heating Energy (J)
HVAC,Sum,Ideal Loads Zone Sensible Cooling Energy (J)
HVAC,Sum,Ideal Loads Zone Latent Cooling Energy (J)
HVAC,Sum,Ideal Loads Zone Total Cooling Energy (J)
HVAC,Sum,Ideal Loads Outdoor Air Sensible Heating Energy (J)
HVAC,Sum,Ideal Loads Outdoor Air Latent Heating Energy (J)
HVAC,Sum,Ideal Loads Outdoor Air Total Heating Energy (J)
HVAC,Sum,Ideal Loads Outdoor Air Sensible Cooling Energy (J)
HVAC,Sum,Ideal Loads Outdoor Air Latent Cooling Energy (J)
HVAC,Sum,Ideal Loads Outdoor Air Total Cooling Energy (J)
HVAC,Sum,Ideal Loads Heat Recovery Sensible Heating Energy (J)
HVAC,Sum,Ideal Loads Heat Recovery Latent Heating Energy (J)
HVAC,Sum,Ideal Loads Heat Recovery Total Heating Energy (J)
HVAC,Sum,Ideal Loads Heat Recovery Sensible Cooling Energy (J)
HVAC,Sum,Ideal Loads Heat Recovery Latent Cooling Energy (J)
HVAC,Sum,Ideal Loads Heat Recovery Total Cooling Energy (J)
HVAC,Average,Ideal Loads Sensible Heating Rate (W)
HVAC,Average,Ideal Loads Latent Heating Rate (W)
HVAC,Average,Ideal Loads Total Heating Rate (W)
HVAC,Average,Ideal Loads Sensible Cooling Rate (W)
HVAC,Average,Ideal Loads Latent Cooling Rate (W)
HVAC,Average,Ideal Loads Total Cooling Rate (W)
HVAC,Average,Ideal Loads Zone Sensible Heating Rate (W)
HVAC,Average,Ideal Loads Zone Latent Heating Rate (W)
HVAC,Average,Ideal Loads Zone Total Heating Rate (W)
HVAC,Average,Ideal Loads Zone Sensible Cooling Rate (W)
HVAC,Average,Ideal Loads Zone Latent Cooling Rate (W)
HVAC,Average,Ideal Loads Zone Total Cooling Rate (W)

HVAC,Average,Ideal Loads Outdoor Air Sensible Heating Rate (W)
HVAC,Average,Ideal Loads Outdoor Air Latent Heating Rate (W)
HVAC,Average,Ideal Loads Outdoor Air Total Heating Rate (W)
HVAC,Average,Ideal Loads Outdoor Air Sensible Cooling Rate (W)
HVAC,Average,Ideal Loads Outdoor Air Latent Cooling Rate (W)
HVAC,Average,Ideal Loads Outdoor Air Total Cooling Rate (W)
HVAC,Average,Ideal Loads Heat Recovery Sensible Heating Rate (W)
HVAC,Average,Ideal Loads Heat Recovery Latent Heating Rate (W)
HVAC,Average,Ideal Loads Heat Recovery Total Heating Rate (W)
HVAC,Average,Ideal Loads Heat Recovery Sensible Cooling Rate (W)
HVAC,Average,Ideal Loads Heat Recovery Latent Cooling Rate (W)
HVAC,Average,Ideal Loads Heat Recovery Total Cooling Rate (W)
HVAC,Sum,Ideal Loads Time Economizer Active (hr)
HVAC,Sum,Ideal Loads Time Heat Recovery Active (hr)
HVAC,Sum,Max SimAir Iterations ()
HVAC,Sum,Tot SimAir Iterations ()
HVAC,Sum,Tot SimAirLoopComponents Calls ()
HVAC,Sum,Time Zone Temperature Oscillating (hr)
HVAC,Sum,Time Any Zone Temperature Oscillating (hr)
Zone,Sum,Time Not Comfortable Summer Clothes (hr)
Zone,Sum,Time Not Comfortable Winter Clothes (hr)
Zone,Sum,Time Not Comfortable Summer Or Winter Clothes (hr)
Zone,Sum,Time Not Comfortable Summer Clothes Any Zone (hr)
Zone,Sum,Time Not Comfortable Winter Clothes Any Zone (hr)
Zone,Sum,Time Not Comfortable Summer Or Winter Clothes Any Zone (hr)
Zone,Sum,Time Heating Setpoint Not Met (hr)
Zone,Sum,Time Heating Setpoint Not Met While Occupied (hr)
Zone,Sum,Time Cooling Setpoint Not Met (hr)
Zone,Sum,Time Cooling Setpoint Not Met While Occupied (hr)
Zone,Sum,Time Heating Setpoint Not Met Any Zone (hr)
Zone,Sum,Time Cooling Setpoint Not Met Any Zone (hr)
Zone,Sum,Time Heating Setpoint Not Met While Occupied Any Zone (hr)
Zone,Sum,Time Cooling Setpoint Not Met While Occupied Any Zone (hr)
Zone,Average,Schedule Value ()
HVAC,Average,System Node Temp (C)
HVAC,Average,System Node MassFlowRate (kg/s)

HVAC,Average,System Node Humidity Ratio (kgWater/kgDryAir)

HVAC,Average,System Node Setpoint Temp (C)

HVAC,Average,System Node Setpoint Temp Hi (C)

HVAC,Average,System Node Setpoint Temp Lo (C)

HVAC,Average,System Node Setpoint Humidity Ratio (kgWater/kgDryAir)

HVAC,Average,System Node Setpoint Humidity Ratio Min (kgWater/kgDryAir)

HVAC,Average,System Node Setpoint Humidity Ratio Max (kgWater/kgDryAir)

HVAC,Average,System Node Relative Humidity (%)

HVAC,Average,System Node Pressure (Pa)

HVAC,Average,System Node Volume Flow Rate Standard Density (m3/s)

HVAC,Average,System Node Volume Flow Rate Current Density (m3/s)

HVAC,Average,System Node Current Density (kg/m3)

HVAC,Average,System Node Enthalpy (J/kg)

HVAC,Average,System Node Wetbulb Temp (C)

HVAC,Average,System Node Dewpoint Temperature (C)

HVAC,Average,System Node Quality ()

HVAC,Average,System Node Height (m)

HVAC,Sum,Carbon Equivalent Pollution From NOx (kg)

HVAC,Sum,Carbon Equivalent Pollution From CH4 (kg)

HVAC,Sum,Carbon Equivalent Pollution From CO2 (kg)

HVAC,Sum,Zone Mechanical Ventilation No Load Heat Removal (J)

HVAC,Sum,Zone Mechanical Ventilation Cooling Load Increase (J)

HVAC,Sum,Zone Mech Ventilation Cooling Load Increase: OverHeating (J)

HVAC,Sum,Zone Mechanical Ventilation Cooling Load Decrease (J)

HVAC,Sum,Zone Mechanical Ventilation No Load Heat Addition (J)

HVAC,Sum,Zone Mechanical Ventilation Heating Load Increase (J)

HVAC,Sum,Zone Mech Ventilation Heating Load Increase: OverCooling (J)

HVAC,Sum,Zone Mechanical Ventilation Heating Load Decrease (J)

HVAC,Average,Zone Mechanical Ventilation Mass Flow Rate (kg/s)

HVAC,Sum,Zone Mechanical Ventilation Mass (kg)

HVAC,Average,Zone Mechanical Ventilation Volume Flow Rate Standard Density (m3/s)

HVAC,Sum,Zone Mechanical Ventilation Volume Standard Density (m3)

HVAC,Average,Zone Mechanical Ventilation Volume Flow Rate Current Density (m3/s)

HVAC,Sum,Zone Mechanical Ventilation Volume Current Density (m3)

HVAC,Average,Zone Mechanical Ventilation Air Change Rate (ach)

```
Var Type (reported time step),Var Report Type,Variable Name (Units)
Zone,Meter,EnergyTransfer: Facility (J)
Zone,Meter,EnergyTransfer: Building (J)
Zone,Meter,EnergyTransfer: Zone: SPACE_1 (J)
Zone,Meter,Heating: EnergyTransfer (J)
Zone,Meter,Heating: EnergyTransfer: Zone: SPACE_1 (J)
Zone,Meter,Cooling: EnergyTransfer (J)
Zone,Meter,Cooling: EnergyTransfer: Zone: SPACE_1 (J)
Zone,Meter,DistrictHeating: Facility (J)
Zone,Meter,DistrictHeating: HVAC (J)
Zone,Meter,Heating: DistrictHeating (J)
Zone,Meter,DistrictCooling: Facility (J)
Zone,Meter,DistrictCooling: HVAC (J)
Zone,Meter,Cooling: DistrictCooling (J)
Zone,Meter,Carbon Equivalent: Facility (kg)
Zone,Meter,CarbonEquivalentEmissions: Carbon Equivalent (kg)
```

Appendix C

Sample GP Trees

The best GP tree of Melbourne and Las Vegas from chapter 6 are given here. The reason we chose Melbourne and Las Vegas, is that these two cities have the best and the worst annual energy consumption.

Melbourne best model:

(addroot1 (addcube1 (maximum1 (maximum1 (division (maximum1 0.23802075925412114 (average1 0.5982606074906827 0.5794650742572466)) (maximum1 (average1 0.5982606074906827 0.5794650742572466) (average1 0.5982606074906827 0.5794650742572466))) 0.9581037091100292) (average1 0.5982606074906827 0.5794650742572466)) (division (maximum1 (maximum1 0.23802075925412114 (average1 0.5982606074906827 0.5794650742572466)) (average1 (multipli 0.6461452492764632 (maximum1 (multipli 0.22502028343090164 0.5723226040735515) (multipli (division (if-else1 (minimum1 0.8332626279492678 0.18201361502188362) (maximum1 0.24598146022159417 0.6562232128433211) (if-else1 (division (halfback (multipli (division 0.8912052179555682 0.7210459370504537) (halfback (average1 0.9734107084151017 0.9315412886989417)))) 0.27983328719349065) 0.11803854671083824 0.7092480044421522 0.7201152202547952) (minimum1 0.09487320468019356 0.48426699534643014)) (addroot1 (addcube1 (maximum1 (maximum1 (division (maximum1 0.23802075925412114 (average1 0.5982606074906827 0.5794650742572466)) (maximum1 (average1 0.5982606074906827 0.5794650742572466) (average1 0.5982606074906827 0.5794650742572466))) 0.9581037091100292) (average1 0.5982606074906827 0.5794650742572466)) (division (maximum1 (maximum1 0.23802075925412114 (average1 0.5982606074906827 0.5794650742572466)) (average1 (multipli 0.6461452492764632 (maximum1 (multipli 0.22502028343090164 0.5723226040735515) (multipli (division (if-

else1 (minimum1 0.8332626279492678 0.18201361502188362) (maximum1
0.24598146022159417 0.6562232128433211) (if-else1 (division (halfback (mul-
tipli (division 0.8912052179555682 0.7210459370504537) (halfback (aver-
age1 0.9734107084151017 0.9315412886989417)))) 0.27983328719349065)
0.11803854671083824 0.7092480044421522 0.7201152202547952) (mini-
mum1 0.09487320468019356 0.48426699534643014)) (average1 (multipli
0.5176163365664596 0.07082877710090374) 0.7200782109113433)) (halfback (maxi-
mum1 (multipli 0.8350105786074632 0.7530275754696014) 0.5982606074906827)))))
0.5723226040735515)) (average1 0.5982606074906827 0.5794650742572466))
0.018981131922273353 (firstflo (flexfrgr 580491.0 764817.0 495380.0 (adddoor1 (half-
back (multipli 0.032934826817695884 0.5723226040735515)) 0.5794650742572466
644463.0 454719.0) (addwindo (maximum1 0.23802075925412114 (halfforw (half-
back (minimum1 (halfback (division 0.9006609150780807 0.30324072715874484))
(halfforw (minimum1 0.8837478143120694 (halfback (average1 0.9734107084151017
0.9315412886989417)))))))) (halfforw 0.06343084569329926) 601135.0) 951274.0)
(flexgrid 768294.0 801935.0 (windover (if-else1 (maximum1 0.8239265926840986
(if-else1 (maximum1 (multipli (halfforw 0.5362969437354441) (minimum1
0.4198720369493889 0.949985025208641)) (multipli (maximum1 0.8850354056565451
0.615298739405795) (multipli 0.5458108600345801 0.9289501990655119))) (halfforw
0.5029139619675101) (maximum1 (halfback (halfforw 0.32756675036822747)) (mul-
tipli (minimum1 0.9641896990858216 0.23061921056525114) 0.5113984693984678))
(maximum1 (maximum1 (halfforw 0.7138749269815114) (halfforw
0.15953365407582165)) 0.6881161317626318))) 0.04191529329834187 (division
(halfforw 0.5436653364549653) (division 0.6097066681074366 0.49838234988574226))
(minimum1 (halfforw 0.7894653339157164) 0.22005427363054642)) (halfforw (mul-
tipli (halfforw 0.18977788136355211) 0.5982606074906827)) 0.7285496118508177
(division (average1 (minimum1 0.34033375713450476 0.2734537948371105) (multipli
(if-else1 (if-else1 0.696193444139585 0.46927288357504426 0.5322541363601059
0.7776399778622556) 0.9185813911262478 (maximum1 0.27093718850541815
0.37888764452999846) (maximum1 0.020261202074310924 0.7116371823161038))
0.49686348683519344)) 0.039643768928812384) 0.886880133916103 (decre-
men 487736.0)) 629844.0) (flexgrid 768294.0 801935.0 (addwindo (aver-
age1 0.5982606074906827 (minimum1 (maximum1 (division (minimum1
0.09487320468019356 0.48426699534643014) 0.5881663238953361) (halfforw
(division (maximum1 0.23802075925412114 (average1 0.5982606074906827
0.5794650742572466)) 0.5794650742572466))) (halfback (average1 (average1

0.5982606074906827 0.5794650742572466) 0.9315412886989417)))) (maximum1 0.23802075925412114 (average1 0.5982606074906827 0.5794650742572466)) 601135.0) 629844.0) (flexgrid 629844.0 (average2 149912.0 951355.0) (addwindo (minimum1 (maximum1 (division 0.6152907292534696 0.5881663238953361) (half-forw (division (maximum1 0.23802075925412114 (average1 0.5982606074906827 0.5794650742572466)) (average1 0.5982606074906827 0.5794650742572466)))) (halfback (average1 0.9734107084151017 0.9315412886989417))) (halfforw (half-back (minimum1 (halfback (division 0.9006609150780807 0.30324072715874484)) (halfforw (division (halfback (multipli (halfback (average1 0.4855462345618463 0.7038022919284811)) (halfforw (halfback (minimum1 (halfback (division 0.9006609150780807 0.30324072715874484)) (halfforw (division (halfback 0.6386790667211367) 0.27983328719349065))))))) 0.27983328719349065))))) (if-else2 (incremen (if-else2 (incremen (minimum2 327552.0 199169.0)) (if-else2 (incremen 857116.0) (if-else2 178906.0 10899.0 365424.0 521590.0) (average2 626474.0 846376.0) (decremen 134013.0)) (incremen (minimum2 850943.0 747154.0)) (incremen 22620.0))) (if-else2 (incremen 857116.0) (if-else2 178906.0 10899.0 365424.0 521590.0) 629844.0 (if-else2 (average2 (if-else2 (if-else2 779063.0 (incremen (max-imum2 628739.0 140956.0)) (decremen (minimum2 985173.0 363100.0)) (average2 (maximum2 52957.0 25487.0) (if-else2 196767.0 706375.0 517986.0 409332.0))) (decremen (maximum2 (if-else2 469800.0 151393.0 123220.0 380105.0) (minimum2 548546.0 234477.0))) 629844.0 (average2 (incremen (average2 485862.0 259330.0)) (decremen (if-else2 111450.0 757153.0 513737.0 137301.0)))) (minimum2 (average2 190940.0 92631.0) (minimum2 746581.0 661009.0))) 752903.0 (incremen (mini-mum2 (if-else2 (if-else2 178906.0 10899.0 365424.0 521590.0) (if-else2 178906.0 10899.0 365424.0 521590.0) (incremen 427240.0) (if-else2 (average2 (decremen (min-imum2 658780.0 529546.0)) (minimum2 (minimum2 (average2 190940.0 92631.0) (minimum2 746581.0 661009.0)) (minimum2 746581.0 661009.0))) 752903.0 (incre-men (if-else2 674278.0 258984.0 197702.0 336633.0)) (if-else2 (if-else2 (decremen 886420.0) (minimum2 328484.0 590821.0) (minimum2 920568.0 985272.0) (max-imum2 14062.0 161516.0)) (minimum2 (average2 241861.0 216777.0) (decremen 754678.0)) 801935.0 (incremen (if-else2 674278.0 258984.0 197702.0 529671.0))))) 199169.0)) (if-else2 (if-else2 (decremen 327552.0) (minimum2 328484.0 590821.0) (minimum2 920568.0 985272.0) (maximum2 14062.0 161516.0)) (minimum2 (av-erage2 241861.0 216777.0) (decremen 754678.0)) 801935.0 (incremen (if-else2 674278.0 258984.0 197702.0 336633.0))))) (incremen (if-else2 (incremen (mini-mum2 327552.0 199169.0)) (if-else2 (incremen 857116.0) (if-else2 178906.0 10899.0

365424.0 521590.0) (average2 626474.0 846376.0) (decremen 134013.0)) (incremen (minimum2 850943.0 747154.0)) (incremen 22620.0))) (incremen 22620.0))) 629844.0) 17458.0 965155.0)) (simplero (decremen 17458.0)))(average1 (multipli 0.5176163365664596 0.07082877710090374) 0.7200782109113433)) (halfback (maximum1 (multipli 0.8350105786074632 0.7530275754696014) 0.5982606074906827))))) 0.5723226040735515)) (average1 0.5982606074906827 0.5794650742572466)) 0.018981131922273353 (firstflo (flexfrgr 580491.0 764817.0 495380.0 (adddoor1 (halfback (multipli 0.032934826817695884 0.5723226040735515)) 0.5794650742572466 644463.0 454719.0) (addwindo (maximum1 0.23802075925412114 (halfforw (halfback (minimum1 (halfback (division 0.9006609150780807 0.30324072715874484)) (halfforw (minimum1 0.8837478143120694 (halfback (average1 0.9734107084151017 0.9315412886989417)))))))) (halfforw 0.06343084569329926) 601135.0) 951274.0) (flexgrid 768294.0 801935.0 (windover (if-else1 (maximum1 0.8239265926840986 (if-else1 (maximum1 (multipli (halfforw 0.5362969437354441) (minimum1 0.4198720369493889 0.949985025208641)) (multipli (maximum1 0.8850354056565451 0.615298739405795) (multipli 0.5458108600345801 0.9289501990655119))) (halfforw 0.5029139619675101) (maximum1 (halfback (halfforw 0.32756675036822747)) (multipli (minimum1 0.9641896990858216 0.23061921056525114) 0.5113984693984678)) (maximum1 (maximum1 (halfforw 0.7138749269815114) (halfforw 0.15953365407582165)) 0.6881161317626318))) 0.04191529329834187 (division (halfforw 0.5436653364549653) (division 0.6097066681074366 0.49838234988574226)) (minimum1 (halfforw 0.7894653339157164) 0.22005427363054642)) (halfforw (multipli (halfforw 0.18977788136355211) 0.5982606074906827)) 0.7285496118508177 (division (average1 (minimum1 0.34033375713450476 0.2734537948371105) (multipli (if-else1 (if-else1 0.696193444139585 0.46927288357504426 0.5322541363601059 0.7776399778622556) 0.9185813911262478 (maximum1 0.27093718850541815 0.37888764452999846) (maximum1 0.020261202074310924 0.7116371823161038)) 0.49686348683519344)) 0.039643768928812384) 0.886880133916103 (decremen 487736.0)) 629844.0) (flexgrid 768294.0 801935.0 (addwindo (average1 0.5982606074906827 (minimum1 (maximum1 (division (minimum1 0.09487320468019356 0.48426699534643014) 0.5881663238953361) (halfforw (division (maximum1 0.23802075925412114 (average1 0.5982606074906827 0.5794650742572466)) 0.5794650742572466))) (halfback (average1 (average1 0.5982606074906827 0.5794650742572466) 0.9315412886989417)))) (maximum1 0.23802075925412114 (average1 0.5982606074906827 0.5794650742572466)) 601135.0) 629844.0) (flexgrid 629844.0 (average2 149912.0 951355.0) (addwindo (minimum1 (maximum1 (division 0.6152907292534696 0.5881663238953361) (halfforw (division (maximum1 0.23802075925412114 (average1 0.5982606074906827 0.5794650742572466)) (average1 0.5982606074906827 0.5794650742572466)))) (halfback (average1 0.9734107084151017 0.9315412886989417))) (halfforw (halfback (minimum1 (halfback (division 0.9006609150780807 0.30324072715874484)) (halfforw (division (halfback (multipli (halfback (average1 0.4855462345618463 0.7038022919284811)) (halfforw (halfback (minimum1 (halfback (division 0.9006609150780807 0.30324072715874484)) (halfforw (division (halfback 0.6386790667211367) 0.27983328719349065))))))) 0.27983328719349065))))) (if-else2 (incremen (if-else2 (incremen (minimum2 327552.0 199169.0)) (if-else2 (incremen 857116.0) (if-else2 178906.0 10899.0 365424.0 521590.0) (average2 626474.0 846376.0) (decremen 134013.0)) (incremen (minimum2 850943.0 747154.0)) (incremen 22620.0))) (if-else2 (incremen 857116.0) (if-else2 178906.0 10899.0 365424.0 521590.0) 629844.0 (if-else2 (average2 (if-else2 (if-else2 779063.0 (incremen (maximum2 628739.0 140956.0)) (decremen (minimum2 985173.0 363100.0)) (average2 (maximum2 52957.0 25487.0) (if-else2 196767.0 706375.0 517986.0 409332.0))) (decremen (maximum2 (if-else2 469800.0 151393.0 123220.0 380105.0) (minimum2 548546.0 234477.0))) 629844.0 (average2 (incremen (average2 485862.0 259330.0)) (decremen (if-else2 111450.0 757153.0 513737.0 137301.0)))) (minimum2 (average2 190940.0 92631.0) (minimum2 746581.0 661009.0))) 752903.0 (incremen (minimum2 (if-else2 (if-else2 178906.0 10899.0 365424.0 521590.0) (if-else2 178906.0 10899.0 365424.0 521590.0) (incremen 427240.0) (if-else2 (average2 (decremen (minimum2 658780.0 529546.0)) (minimum2 (minimum2 (average2 190940.0 92631.0) (minimum2 746581.0 661009.0)) (minimum2 746581.0 661009.0))) 752903.0 (incremen (if-else2 674278.0 258984.0 197702.0 336633.0)) (if-else2 (if-else2 (decremen 886420.0) (minimum2 328484.0 590821.0) (minimum2 920568.0 985272.0) (maximum2 14062.0 161516.0)) (minimum2 (average2 241861.0 216777.0) (decremen 754678.0)) 801935.0 (incremen (if-else2 674278.0 258984.0 197702.0 529671.0))))) 199169.0)) (if-else2 (if-else2 (decremen 327552.0) (minimum2 328484.0 590821.0) (minimum2 920568.0 985272.0) (maximum2 14062.0 161516.0)) (minimum2 (average2 241861.0 216777.0) (decremen 754678.0)) 801935.0 (incremen (if-else2 674278.0 258984.0 197702.0 336633.0))))) (incremen (if-else2 (incremen (minimum2 327552.0 199169.0)) (if-else2 (incremen 857116.0) (if-else2 178906.0 10899.0 365424.0 521590.0) (average2 626474.0 846376.0) (decremen 134013.0)) (incremen (minimum2 850943.0 747154.0)) (incremen 22620.0))) (incremen 22620.0))) 629844.0) 17458.0 965155.0)) (simplero (decremen 17458.0)))

Las Vegas best model:

(addroot1 (addcube1 (if-else1 0.5434755413282031 0.11145929170228419 0.9929647757941246 0.5623001629234392) (if-else1 (multipli 0.18201659170105422 (minimum1 (average1 (division 0.18201659170105422 (division 0.16202652043228216 0.6236854701814866)) (if-else1 (division 0.4723927466199672 0.5306001095042617) (average1 0.90909151182931 0.7659181018563209) (halfforw 0.02211462470345904) 0.2164138408961228)) (if-else1 (if-else1 0.5434755413282031 0.11145929170228419 0.9929647757941246 0.5623001629234392) (if-else1 (halfforw 0.9265661120857922) (division 0.2576355253447169 0.823184066379223) (maximum1 (halfforw 0.4010723049911942) 0.652338713798544) (multipli 0.7202752534334543 0.9062024949020988)) (if-else1 (division 0.9754253513262582 0.03270497923065263) (halfforw 0.23862013425627882) (maximum1 0.7980563860352208 0.3155275702336391) (minimum1 0.948947082510833 0.2669393522387139)) (half-forw (if-else1 (average1 (minimum1 (maximum1 (halfforw 0.4010723049911942) 0.30503964312686715) (halfforw (if-else1 0.9347956302354887 0.7175022634546976 0.012969342153151375 0.773709698277249))) (halfback 0.5901461046496381)) 0.8493519238615178 0.2051605870899571 0.9062024949020988))))) 0.11393235288137582 (average1 0.07319330978483485 0.7320246699378354) (if-else1 0.5434755413282031 0.11145929170228419 0.9929647757941246 0.5623001629234392)) (if-else1 (maximum1 0.302384900129143 0.9444115278190125) 0.03270497923065263 0.9834055685941071 (halfback 0.5399259484936939)) (firstflo (flexfrgr (average2 707254.0 626405.0) 342899.0 676791.0 (ad-ddoor1 0.8197694559669279 (if-else1 (division (average1 0.823184066379223 0.3472208351567573) (if-else1 0.18056322679352632 0.4336921047532356 (maximum1 0.302384900129143 0.9444115278190125) 0.20586920548911825)) (if-else1 (maximum1 0.5562343471015694 0.33944364191015486) (division 0.2576355253447169 0.823184066379223) (average1 0.82804989 0.48673320) (multipli 0.7202752534334543 0.9062024949020988)) (if-else1 (division 0.9754253513262582 (maximum1 (halfforw 0.03270497923065263) 0.8464034463063537)) (half-forw (division (maximum1 0.2164138408961228 0.10679972917992153) 0.9558371596130881)) (maximum1 0.11145929170228419 0.3155275702336391) (minimum1 0.948947082510833 0.2669393522387139)) (halfforw (if-else1 0.9157197009145568 0.8493519238615178 (maximum1 0.302384900129143 0.9444115278190125) 0.24944184766947441))) 706578.0 717308.0) (addwindo 0.9834055685941071 (halfforw (if-else1 0.5434755413282031 0.11145929170228419 0.9929647757941246 0.5623001629234392)) (minimum2 648165.0 (average2 25425.0

931758.0))) 346469.0) (flexgrid (minimum2 986092.0 (average2 25425.0 931758.0)) 279246.0 (addwindo (maximum1 0.3622638331517275 0.30503964312686715) (maximum1 (halfforw 0.4010723049911942) 0.652338713798544) (minimum2 986092.0 (average2 25425.0 931758.0))) (if-else2 (maximum2 (incremen (average2 (decremen 127708.0) (maximum2 (incremen (average2 (incremen 463375.0) 11658.0)) (decremen (decremen (average2 434175.0 606317.0)))))) (decremen (decremen (average2 434175.0 606317.0)))) (incremen 642602.0) (maximum2 (incremen (average2 (decremen 127708.0) 11658.0)) (decremen (decremen (incremen 642602.0)))) (decremen 173710.0))) (flexgrid (incremen 463375.0) 279246.0 (addwindo (maximum1 0.3622638331517275 0.30503964312686715) (maximum1 (halfforw 0.4010723049911942) (if-else1 0.5434755413282031 0.11145929170228419 0.9929647757941246 0.5623001629234392)) (average2 25425.0 931758.0)) (if-else2 (maximum2 (incremen (average2 (decremen 127708.0) 11658.0)) (decremen (decremen (incremen 642602.0)))) (incremen 642602.0) (average2 (decremen 127708.0) 11658.0) (decremen 173710.0))) (flexgrid 790227.0 528906.0 (addwindo (maximum1 0.3622638331517275 0.30503964312686715) (maximum1 (halfforw 0.4010723049911942) (division (if-else1 (if-else1 (multipli 0.18201659170105422 (minimum1 (maximum1 0.5836880270421764 0.6168256711698709) (if-else1 (division (average1 0.44957849557411633 0.3472208351567573) (if-else1 0.18056322679352632 0.4336921047532356 (maximum1 (halfforw 0.4010723049911942) (division 0.2164138408961228 0.9558371596130881)) 0.20586920548911825)) (if-else1 (halfforw 0.9265661120857922) (division 0.2576355253447169 0.823184066379223) (average1 0.8280498942734827 0.48673320827855227) (maximum1 0.776230309060195 0.10679972917992153)) 0.8464034463063537 (halfforw (if-else1 0.9157197009145568 0.8493519238615178 0.2051605870899571 0.24944184766947441))))) (division (halfforw 0.4010723049911942) 0.6236854701814866) (average1 0.07319330978483485 0.7320246699378354) (if-else1 0.5434755413282031 0.11145929170228419 0.9929647757941246 0.5623001629234392)) (if-else1 (halfforw 0.9265661120857922) (division 0.2576355253447169 0.823184066379223) (if-else1 (halfforw 0.9265661120857922) (division 0.2576355253447169 0.823184066379223) (average1 0.8280498942734827 0.48673320827855227) (maximum1 0.776230309060195 0.10679972917992153)) (multipli 0.7202752534334543 0.9062024949020988)) (average1 0.90909151182931 0.7659181018563209) (halfforw (if-else1 0.9157197009145568 0.8493519238615178 0.2051605870899571 0.24944184766947441))) 0.9558371596130881)) (minimum2 648165.0 (average2 25425.0 931758.0))) 590719.0))) (simplero (average2 43622.0 667055.0)))