AS500212

HVAC System Selection with Generative Design

Sean Fruin
Sigma AEC Solutions

Learning Objectives

- Gain an understanding of what generative design is and where it can be utilized.
- Identify how to define rules and measure success for generative design analysis.
- Learn how to use Autodesk's newest generative design program.
- Explore how computational design techniques can solve optimization found in MEP System design.

Description

Selecting the best HVAC system for a given building has been impossible. Why? Because each design is measured with several contradicting metrics. A holistic set of rules can be leveraged to find a range of potential design solutions with generative design. The computers then generate and evaluate a vast number of designs within that range. These solutions can then be presented to a decision-maker, where pros and cons can be objectively and quickly visualized and measured. This session will explore the idea of using generative design to evaluate different mechanical systems for a given building. We will examine methods for gathering Revit's geometry and defining parametric rules for other system arrangements. Next, we will take things to the next level by building a Generative Design in Revit workflow that evaluates each design option. Architects and engineers will leave inspired and understand the generative design buzz and leverage this new way of thinking to disrupt the current design process.

Speaker

Sean Fruin is a Mechanical Engineer (EIT), design technologist, and innovator who has an intense fascination with automation and the exploration of computational design solutions for the AEC industry. He has had the opportunity to learn many aspects of the design industry, having worked in manufacturing, MEP designing, and General Contracting. Sean started Sigma AEC Solutions to live his dream, having the opportunity to explore and implement the latest technologies to improve efficiency and increase quality in the AEC industry.

Introduction

At the beginning of 2021, I joined a start-up called iBuilt as a mechanical engineer and computational designer. iBuilt set out to solve the inefficiencies in construction by reimagining what a building is, how it is designed, and how it is constructed. The ship's captain was a real estate developer who focused on cutting costs, guaranteeing clean and stunning façades, and ensuring a revolution in standardized construction methods. These goals came at the expense of safety, legal risks, and actual science and engineering. Being a turbulent start-up, I found myself as the only mechanical engineer in the company with any background in HVAC to push back on this logic. Identifying adequate HVAC systems was one of their top priorities, so I was promptly assigned the most challenging task of my career. I was asked to identify all possible HVAC systems that would work with iBuilt's theoretical modular system. I was required to create a system that was extremely cheap, easily installed, and highly standardized. On top of this challenging task, I was given numerous constraints, such as having infrastructure shafts restricted to stair shafts and not having exterior wall penetrations or ducts.

Energized by the challenge, I dove in, trying to make the vision a reality. With each passing day, more data needed to be organized, additional calculations had to be made, and an increasing number of puzzle pieces emerged. As the reality started to set in that the vision was much more complex than imagined, the process turned taxing. Discussions about building codes or basic engineering devolved into what felt like long debates between lawyers. Conversations often revolved around assumptions with little engineering merit. For instance, the beliefs that using a split system meant that we wouldn't need ducts, that fan selection is independent of duct sizing, and that thermal loads are only dependent on floor area. These assumptions overlooked building code requirements like ducting fresh air into dwelling units and ignored engineering fundamentals like fan selection considerations or equations for calculating cooling loads. For weeks, days were filled with these debates and hearing the same questions asked to multiple engineers/manufacturers in the hope of a different answer. This exhausting uphill battle and frustration with my inability to articulate design reasoning led to burnout and taking a few days off to collect my thoughts and clear my head.

During my time off, I realized that I had been preparing myself for this challenge over the last four years as a computational designer. A traditional design approach would not get the job done, but maybe a generative design methodology would. Working in this data-centric and automated way has many benefits. Formulating a computational system helps organize design logic, allows quick revisions to adapt to design changes, and provides transparency in tradeoffs. Most importantly, developing a generative system would help illustrate that the solution to this puzzle was not a single answer but rather a set of optimal solutions. The process does require a lot of standardized data, and this is what iBuilt did have correct. Autodesk's generative design software was created to tackle these types of multi-objective problems effectively. Through generative design, I was able to overcome my dilemma. This paper lays out how to formulate these generative design workflows and summarizes how the generative design study assesses HVAC systems for iBuilt's modular multifamily buildings.

Optimization

Optimization is at the core of building design. Architects and engineers are responsible for looking at design holistically and identifying the best balance among various objectives. The correct answer often varies between stakeholders. Solving this dynamic tension between contractors, building owners, and regulatory bodies requires a goal-driven approach to evaluate compromises among potential design options. Optimization techniques provide insight into tradeoffs, guiding the design to the best possible solution for all.

Creating optimization problems requires the translation of the design process into mathematical algorithms with three main components. The first component is known as objective functions and provides the means for measuring success. The analysis is done by either maximizing or minimizing these functions. Some examples in construction include:

- **Maximizing rentable floor area**
- **Minimizing equipment noise**
- **Minimizing cost**

The second component is variables that are used to structure, connect, and explore the design space. Variables come in three flavors: static inputs, variable inputs, and parametric variables. For example, a developer trying to maximize profit would have to select an AC that meets the fixed cooling load of the room but could directly choose to use the cheapest AC units on the market. However, that variable indirectly impacts the operating cost of the building due to lower energy efficiency. All these variable types can take several unique forms. A few examples include:

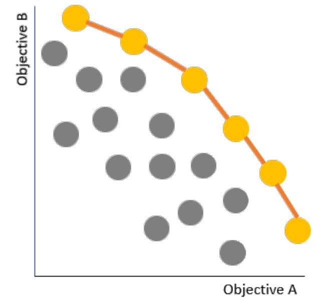
- **Material cost per linear foot**
- **Equipment specifications**
- **Area of a polygon**
- **Location of a point on a line**

Constraints are the last component and represent a limitation on the values of the variables. Adding proper constraints ensures that the solution only includes appropriate values. Obvious constraints include those which directly limit choices:

- **Construction cannot exceed a certain budget**
- **Building code requirements for exhaust air**
- **Piping cannot exceed a specific elevation**

Combining these components forms algorithmic systems capable of generating many design options that repeatedly evolve to better objective function values. When more than one objective function needs to be minimized or maximized, compromises are required on competing objectives. To make an informed decision about what to compromise, we consider alternatives that represent the Pareto optimal solution. At a high level, a Pareto optimal solution is a set of non-inferior solutions in the objective space defining a boundary beyond which none of the objective functions can be improved upon without sacrificing at least one of the other objectives. This idea is a cornerstone concept in the field of multi-objective optimization.

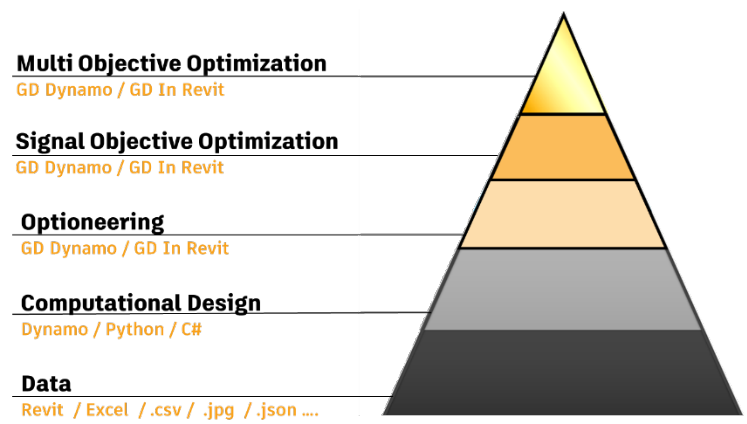
The image to the right illustrates this point. The orange points represent the optimal solution, and all the gray points are suboptimal.



Using this technique clarifies the problem by narrowing down the number of possible solutions and allowing better comparisons between the objectives of the remaining optimal solutions. Autodesk's generative design technology provides a framework for these types of multi-objective design problems. The discussion up to this point has been a bit abstract. The following section breaks down Autodesk's framework to better understand the process and the steps required.

Generative Design Framework

Autodesk's generative design technology makes writing optimization problems for the AEC industry very manageable. Comprehensive generative Design workflows are purely a combination of data, established algorithms, and design knowledge. The challenge for designers is formulating their design knowledge into algorithms that can be employed in the evaluation and optimization process. The best way to produce these workflows is by following a framework to guide the process. Autodesk's journey to create a multiple objective framework has been an ongoing evolution of tools built on top of Revit.

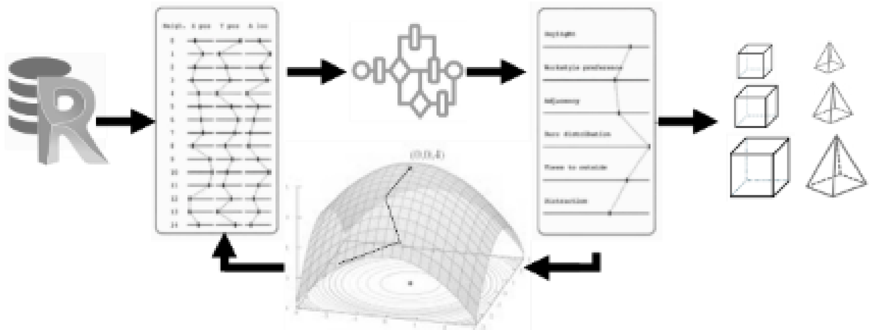


The foundation of a generative design workflow starts with data. This data can take many forms (geometry, equipment specs, building code, labor costs, etc.). While Revit provides a convenient way of storing data, other sources can be used. The key is to ensure the standardization and organization of the data.

The next phase introduced computational/parametric design which provides the means to link all the data, variables, and constraints together to define a design space. The objective is to make a flexible algorithmic system that produces a variety of outputs. Dynamo allows access to Revit data while enabling the ability to manipulate data and geometry in a straightforward visual programming language.

At one time, “optioneering” was done using Project Fractal, which worked okay for smaller projects but not for larger ones. This program allowed the computer to quickly cycle through the design space of the algorithmic system created in Dynamo. The limitations were due to the fact that there was no feedback loop. Because of this, the program had to review the entire design space, variable by variable, to find the optimal solution. As you might imagine, as the number of variable combinations grows, the required calculation time grows exponentially. Due to this, problems with large design spaces took too long for the computer to run.

Today, Revit 2022 ships with a technology that was first called “Project Refinery Beta” but has since been renamed “Generative Design for Revit and Dynamo.” Rather than using brute force to explore the design space, a feedback loop is used to quickly move towards the maximums or minimums of the objective functions. This feedback loop is what makes solving large optimization problems now possible. Specifically, the genetic algorithm - NSGA II is what makes this latest version such a big deal. The process of evolution inspires this optimization algorithm by natural selection, where the variable values from the best solutions are selected for reproduction in an ongoing loop.



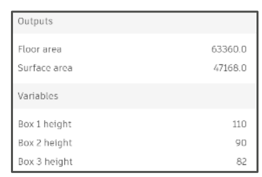
Another feature of “Generative Design for Revit and Dynamo” is the *Explore Outcomes* user interface. The interface provides insight into the outcomes in a few ways. This is especially true for multi-domain, multi-objective optimization problems. The techniques include:

Outcomes

The center pane either displays thumbnails of the geometry generated or a sortable table for each generated outcome.

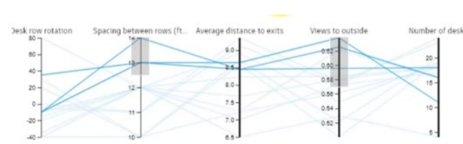
Details

The right pane displays the specific details of a selected outcome that is selected in the center pane. The details include the input variables, generated geometry, and resulting outputs.



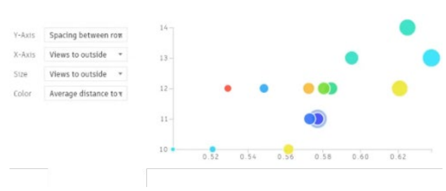
Parallel Coordinates Chart

Parallel coordinates are a visualization technique used to plot different variable combinations across several performance measures. Each variable corresponds to a normalized vertical axis, and each combination is displayed as a series of connected points along the measure/axes. These plots are ideal for comparing many variables together and seeing the relationships between them.



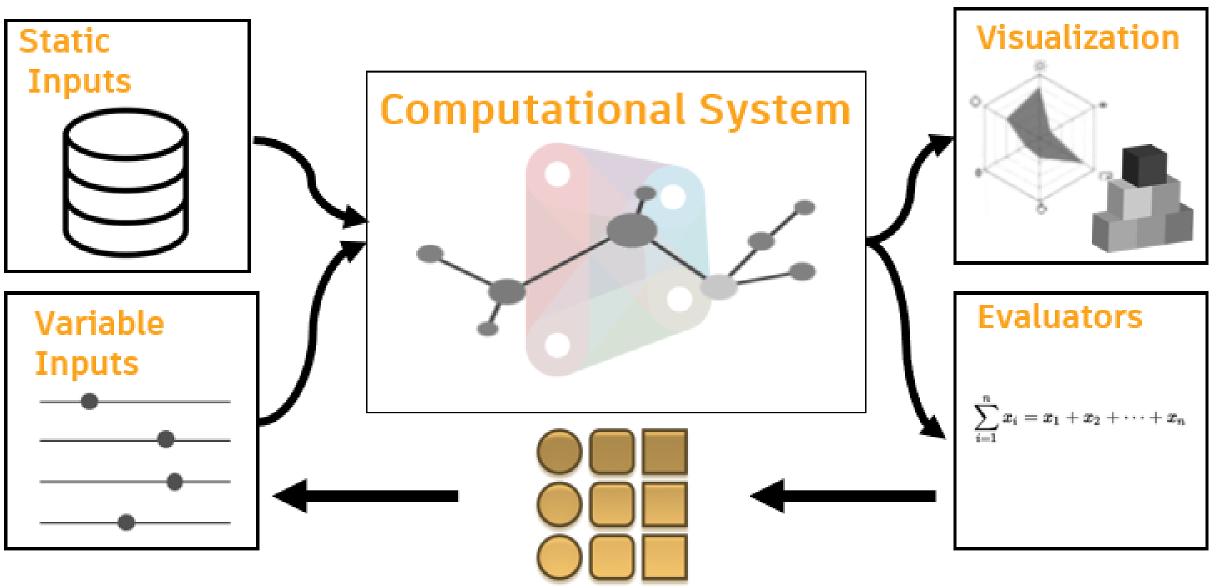
Scatter plots

The scatter plots are a visualization technique used to observe the quality and distribution of solutions between objectives. The scatter plot may only be done in 2D with two variables assigned to the X and Y axes, but variables can also be assigned to the size and color of the dots.



As seen above, all the critical components for formulating and evaluating multiple objective optimization problems have been added to Autodesk's technology over time. This approach to problem-solving is not new but has just started becoming mainstream in the AEC industry in recent years due to Autodesk's efforts to simplify and streamline the process with Dynamo and Generative Design for Revit.

While it has never been easier to harness these emerging technologies, writing algorithms that lead to robust, flexible, and reusable tools is easier said than done. Generative design studies require quite a bit of technical work, adequate upfront data collection, and a complete understanding of the evaluated problem. A framework with the following critical steps guides the process of formulating good studies.



Problem Formation

First, getting an excellent understanding of the problem is critical. Start formulating the problem with the end in mind. Identifying the desired outcomes of a generative design study is crucial because the computer cannot consider any goals that it has not been instructed to consider. Doing this first ensures awareness of what you want to accomplish throughout the process. Next, think about all the steps needed to get from data inputs to measurable outputs. Remember that an ambiguous or incomplete description of the problem leads to poor results. Some questions to help gain insight and define the problem include:

- **What are you trying to design?**
- **What requirements must the design satisfy?**
- **What constraints must the design satisfy?**
- **What defines success or failure of the design?**
- **What are the design parameters?**
- **What is the range for the parameters?**
- **What objective could be maximized or minimized?**

Computational System

Second, map out the series of instructions, rules, and relationships between variables that precisely follow the design process. This step helps break the problem into smaller components by connecting all the direct and indirect variables that govern the design process. Sketches, flowcharts, and mind maps are all wonderful tools that can be applied here. Steps often include the combination of the data, constraints, and simple utility algorithms. Key concepts to consider in this step include:

- **What are the design steps?**
- **What are the design variables?**
- **What are the design constraints?**
- **What equations are involved?**
- **What algorithms can be applied?**
- **What geometry (points, solids, surfaces) are available?**
- **What's the most efficient way to solve each step?**

Identify Variables

In the third step, variables are identified. Naturally, once the design process is mapped out, the variables tend to reveal themselves. Variables can either be static, dynamic, or parametric. For fixed variables, the data source and data structure should be known. For changing variables, the minimum and maximum values need to be set. For parametric variables, the inputs need to be identified. Answers to these questions help determine variables and set their type:

- **What data is available?**
- **What variables are fixed?**
- **What variables can change?**
- **What variables are calculated?**
- **What's the range of the variables?**
- **What are the data structures of the data?**

Objective Functions

The fourth step is to establish the metric for measuring the success of the results. The metrics are calculated with objective functions whose value is to be either maximized or minimized. These functions are used in genetic algorithms to guide simulations towards optimal design solutions. Each solution, therefore, needs a numerical value that indicates how close it came to meeting the overall specifications of the desired solution. These values should produce intuitive results. In other words, the best/worst candidates should harvest the best/worst score values. Establish the objective function by focusing on these questions:

- **How can the goals be measured?**
- **What facts can be measured?**
- **How can the metrics be impacted?**
- **What variables are decision variables?**
- **What changes from the start to the end?**
- **What will be created or manipulated?**

Visualization

With the objective functions locked down, the fifth step is to think about how to best display the results. Geometry can be created inside the Dynamo script to display all kinds of information. Some examples include, text, tables, charts, 2D diagrams, and 3D models. To help identify the best methods, ask the following questions:

- **Do you need 3D visuals, a data table, graphs, or all the above?**
- **Is the graphical outcome in the Explorer Window?**
- **What graphic can be created inside the script?**
- **What graphic best displays the results?**

Results

The last step involves thinking about how to best generate and interpret the results of the generative design study. The optimization tool offers different methods. (Randomized, Optimize, Cross Product, Like This) The key is recognizing the best technique or a combination of techniques to use. Important questions to answer include:

- **How do you want to review the results?**
- **Who is your target audience?**
- **What objective should be focused on?**
- **How many options should be generated?**
- **What do you want to do with the results?**

In reality, the process is generally not linear but more of an iteration through the steps. For starters, do not over-complicate the study by adding too many initial inputs and outputs. It's usually better to start with a high-level algorithm that includes the major parts of a solution. Once worked out, gradually add more detail as you become more aware of design space. This technique of working from a high level to a detailed algorithm is called stepwise refinement. While developing the algorithm, it is important to consider the goldy locks zone of the following components:

simplicity vs density – the algorithm has too little or too much detail.

bias vs. variance – the design space is too small or too large.

complexity vs. continuity – the design space is too flat or too noisy.

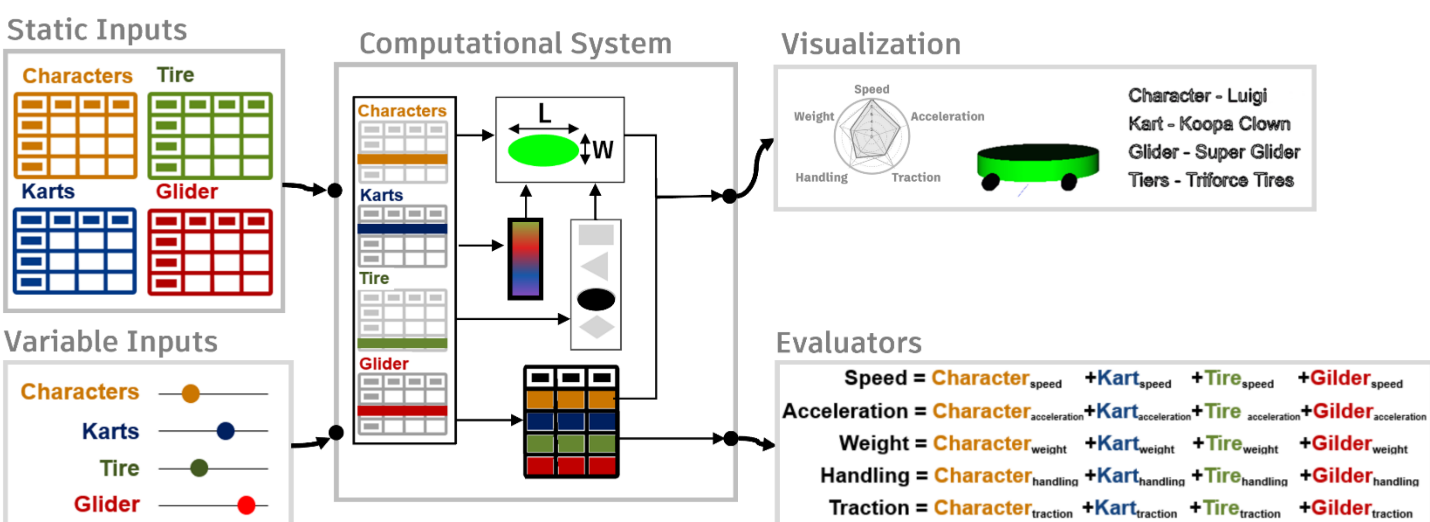
Following the framework outlined in this section helps formulate high-performing generative design studies. The next section demonstrates implementing the framework with a simple multi-objective optimization example.

Optimization - Mario Kart

To illustrate formulating an optimization problem using Autodesk's Generative Design software, a silly, fun, yet straightforward example – "How to maximize your chance of winning at Mario Kart?" In the latest version of this beloved racing game, players select from lists of Nintendo characters, kart frames, tires, and gliders. Combining these options determines the player's performance measured by speed, acceleration, weight, handling, and traction. Being a well-balanced game, there is no one obvious choice that ensures victory. All the performance stats fight with each other. In other words, some combinations maximize the speed while minimizing the acceleration. With 493,920 possible combinations resulting from the 4 inputs, understating the tradeoffs and finding a combination that maximizes the chance of winning is quite tough for a human to determine. On the other hand, a computer can quickly analyze all options and provide the insight necessary to make the right choice for a player's specific goals, play style, and skill level.

Problem Formation

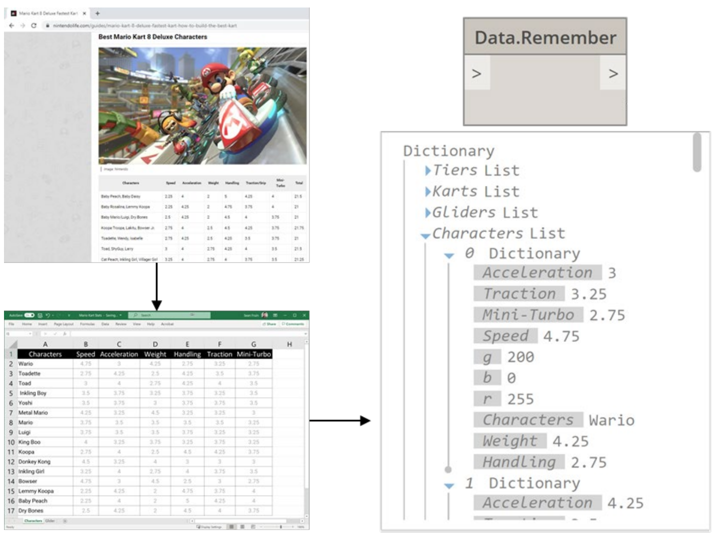
The goal is clear from the problem statement: you wish to maximize your chance of winning at Mario Kart. Translating that goal into measurable objectives is what is required. In this example, the objective functions are built into the game as the player's five performance stats. With a bit of research, each objective function and, more importantly, how the numerical value is derived, can be explained. This graph generates all the possible character, kart frame, tire, and glider combinations by cycling through the data tables of each of the four options using sliders. The resulting speed, acceleration, weight, handling, and traction performance are then calculated. The intention is to identify the best-performing combinations by analyzing the outputs of the generative design study. The high-level flow chart of the study can be seen below.



Static Inputs

The static inputs to this study are the data tables containing character, kart, tire, and glider performance data. All this data needed to be collected and organized. A quick google search got me to the link on the right, where the data tables could be copied and pasted into Excel. Other columns (colors, size, shapes) were added to create our own geometry of the Karts.

Link: Mario Kart 8 Data Tables



The data tables from Excel are imported into Dynamo and converted into dictionaries. The dictionaries get stored inside the Dynamo graph using the **Data.Remember** node. This node ensures generative design studies execute efficiently by caching the input data into the DYN file. The node only accepts geometry (solids, points, meshes, surfaces), strings, and numbers. Revit elements or family types cannot be stored in the node.

Variable Inputs

The variables in this problem are comprised of a player's four options. An **Integer Sliders** node is added to the Dynamo script providing dynamic control over each four-player choice. Each slider needs to be renamed and set as an input. These options are accessed with a left click on the node.

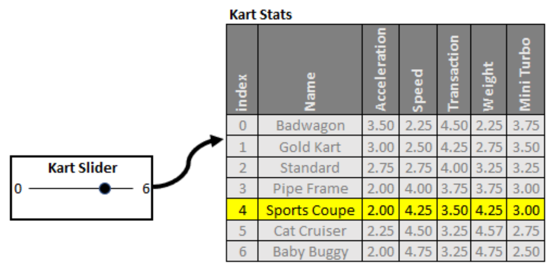
Next, each slider's maximum and minimum value is set by clicking on the arrow in the lower-left corner of the node. These values correspond to the number of options available in each of the players' choices. For example, there are 21 different tire options, so the minimum value is 0, and the maximum value is 20.

Slider Settings

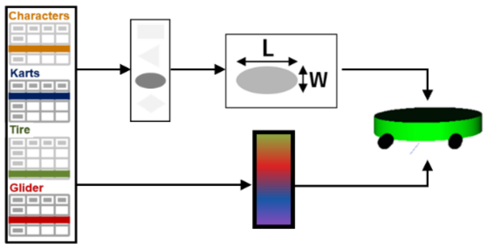
Variable Name	Min	Max	Description
Character	0	41	index of the row in the Character table.
Kart	0	39	index of the row in the Kart table.
Tiers	0	20	index of the row in the Tiers table.
Glider	0	13	index of the row in the Glider table.

Computational System

The computational system has two functions. The generation of options starts by extracting a row from each performance data table (character, kart, tire, and glider). The index of the row comes from the slider values.



Index	Name	Acceleration	Speed	Transaction	Weight	Mini Turbo
0	Badwagon	3.50	2.25	4.50	2.25	3.75
1	Gold Kart	3.00	2.50	4.25	2.75	3.50
2	Standard	2.75	2.75	4.00	3.25	3.25
3	Pipe Frame	2.00	4.00	3.75	3.75	3.00
4	Sports Coupe	2.00	4.25	3.50	4.25	3.00
5	Cat Cruiser	2.25	4.50	3.25	4.57	2.75
6	Baby Buggy	2.00	4.75	3.25	4.75	2.50



Next, geometry is created to represent the build of each kart. Shape, width, length, and color parameters are obtained from the row data and provide the inputs to the kart configurator.

Objective Functions

The objective metrics as described in the game are listed in the table below.

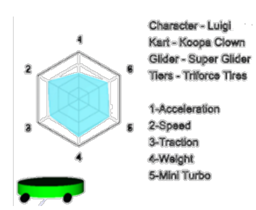
Name	Description
Speed	Top possible speed.
Acceleration	Time to reach your top speed.
Weight	Indicates whether you'll knock someone out of the way or get knocked out.
Handling	How sharply you can drift turn.
Traction	How faster you are on sand and snow, and slip less on ice

Each performance function starts with the character's base stats and adds the values of each of the three parts, just like the game. For example, the equation for the speed is:

$$\textbf{Speed} = \textbf{Character}_{\textbf{speed}} + \textbf{Kart}_{\textbf{speed}} + \textbf{Tier}_{\textbf{speed}} + \textbf{Glider}_{\textbf{speed}}$$

Visualization

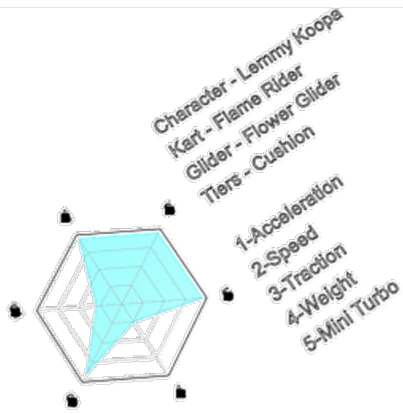
The output of the computational system is Dynamo geometry, representing each unique configuration, a display of the selected components, and a spider graph to show tradeoffs between the objective functions.

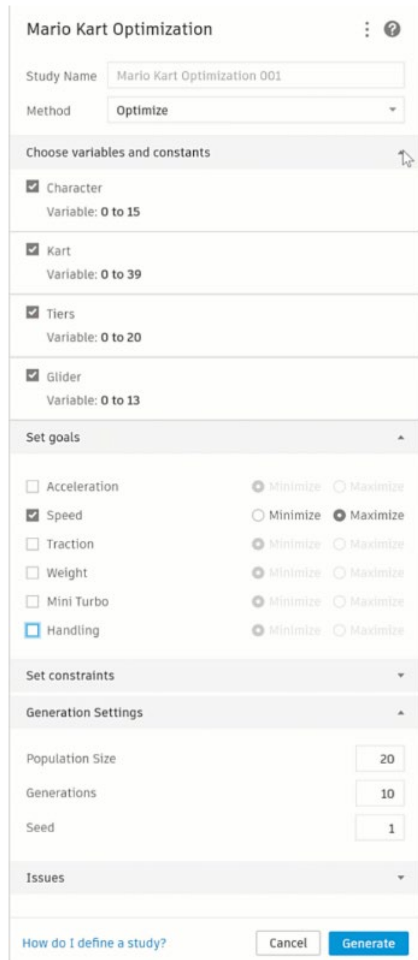


Result

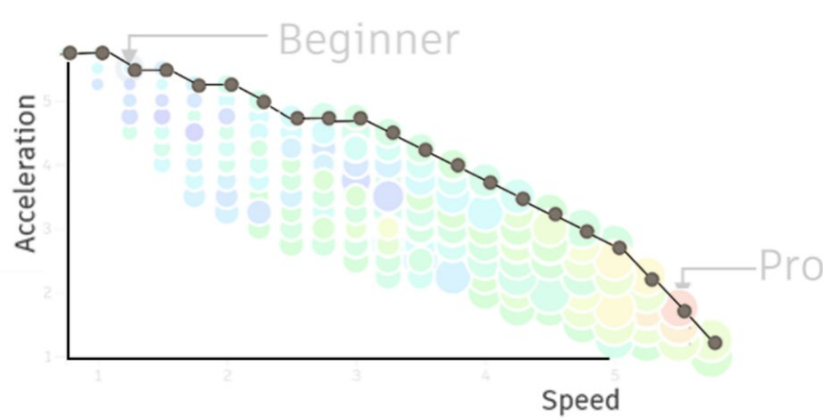
Rather than randomly running through countless options to find a suitable combination, an insightful and comprehensive analysis can quickly be performed using generative design.

By saving the Dynamo script as a generative design study, the exploration can begin. First, let's approach the problem as a single optimization problem where the only goal is to maximize speed. For this study, the optimized method is selected, and the only goal checked is speed. The study rapidly finds the combination with the maximum speed. Looking at the spider graph, it is clear that this combination comes at a massive cost to acceleration.





A better approach to this problem is to utilize the multiobjective abilities of Generative Design. By running the study as a cross-product, a large amount of the design space can be explored, exposing essential insights to maximize the odds of winning by allowing comparisons based on several objectives. Speed and acceleration are generally the two most important attributes in Mario Kart, so the splatter chart below shows those as the X and Y axes for easy comparison. Note that traction and weight are also represented by the color and size of each dot.



The first key observation that can be made is that most of the combinations are bad or suboptimal choices. In fact, optimal configurations between speed and acceleration only make up 5% of all the possible combinations. These optimized solutions all lie on the Pareto Frontier. All of the solutions below this curve are inefficient because you can improve both outcomes with different variables.

The second vital observation is observing the tradeoff. The top-left and bottom-right points indicate the extreme ends of the spectrum where one variable is maximized, spoiling the other. The ideal combination of character, kart, tires, and glider depends on the individual player. For instance, a setup with high acceleration and traction would be best for a new player who may struggle around turns and staying on the road. By contrast, a pro player might be comfortable with low acceleration and high-top speed.

There might not be a combination of variables that permits an easy win; however, there is a more intelligent approach than simply picking kart parts at random and hoping for the best. Most of the 493,920 unique combinations with generative design can be eliminated, leaving 15 optimal combinations tailored to suit the player's style, skill, and the specific race track.

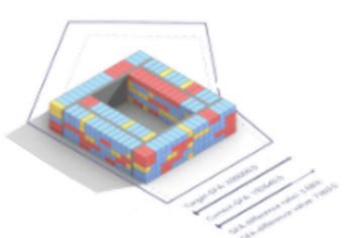
HVAC system selection follows many of the same principles as selecting the optimal MarioKart combination. Just like MarioKart players, engineers have to choose the best mixture of variables that form a seemingly overwhelming number of combinations. Rather than selecting characters, kart frames, tires, and gliders, the engineer selects different system types, varying equipment locations, routing options, and construction methods. Like the game, these options determine the performance, and a generative design study can be formulated to gain insight into the best options.

Optimization – HVAC System

The previous sections have presented computational approaches, techniques, and tools for analyzing multi-objective problems using Generative Design for Dynamo. This section builds on this framework to analyze different HVAC systems for modular multi-family buildings.

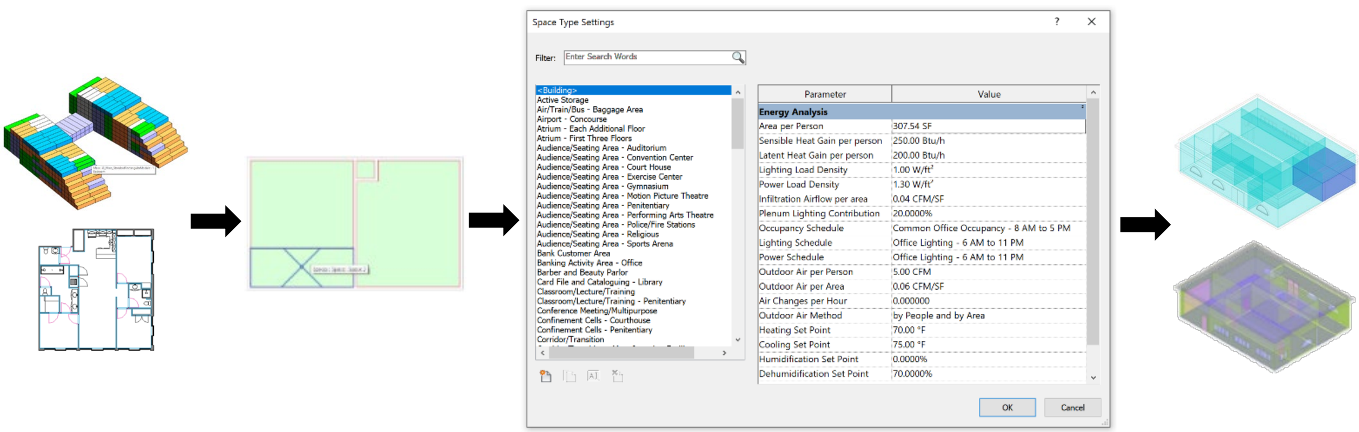
HVAC systems selection requires a tradeoff between upfront costs, life cycle costs, environmental impact, and other factors. The challenge is finding the best-suited system that meets project constraints and stakeholders' priorities. Historically, this process has taken a top-down approach where knowledgeable senior engineers rely less on raw data and more on experience or rules of thumb. This example uses a bottom-up approach where data and automation drive the process.

As mentioned, the foundation for automation is the collection, organization, and centralization of data. iBuilt was centered around this critical idea making it possible to fully utilize these new technologies. Data gathering starts with iBuilt's architecture process that combines a building configurator, kit of parts, and unit plans. This highly standardized and data-centric process generates intelligent architecture Revit models that contain all the project-specific architecture data required to engineer an HVAC System. This data incorporates Revit rooms, room names, unit identification, building thermal properties, and geometry for infrastructure routing.

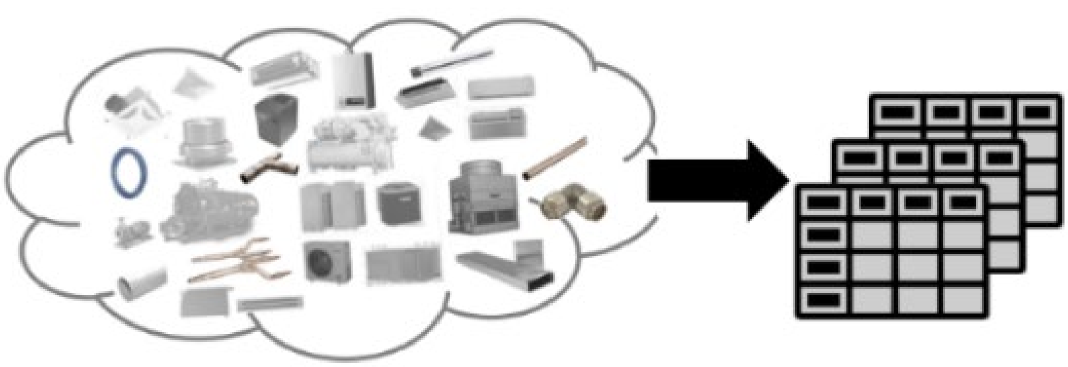


Additionally, the standardized architectural model leads to quick and successful energy modeling. In the mechanical Revit model, the linked architecture rooms are converted into Revits Spaces.

Spaces then get assigned a Space Type which are used to assign internal loads, occupancy and lighting schedules, and ventilation rates. Next, an energy model is automatically created by combining the linked mass elements, linked architectural elements, and Spaces. The energy model generates a .gbxml file, Revit Analytical Spaces, and Revit Analytical Surfaces. After the successful creation of the energy model, the Systems Analysis tool is run using the defualt “load calculation” workflow. The workflow translates the Revit’s .gbxml to EnergyPlus and automatically adds the thermal sizing loads to the Analytical Spaces.

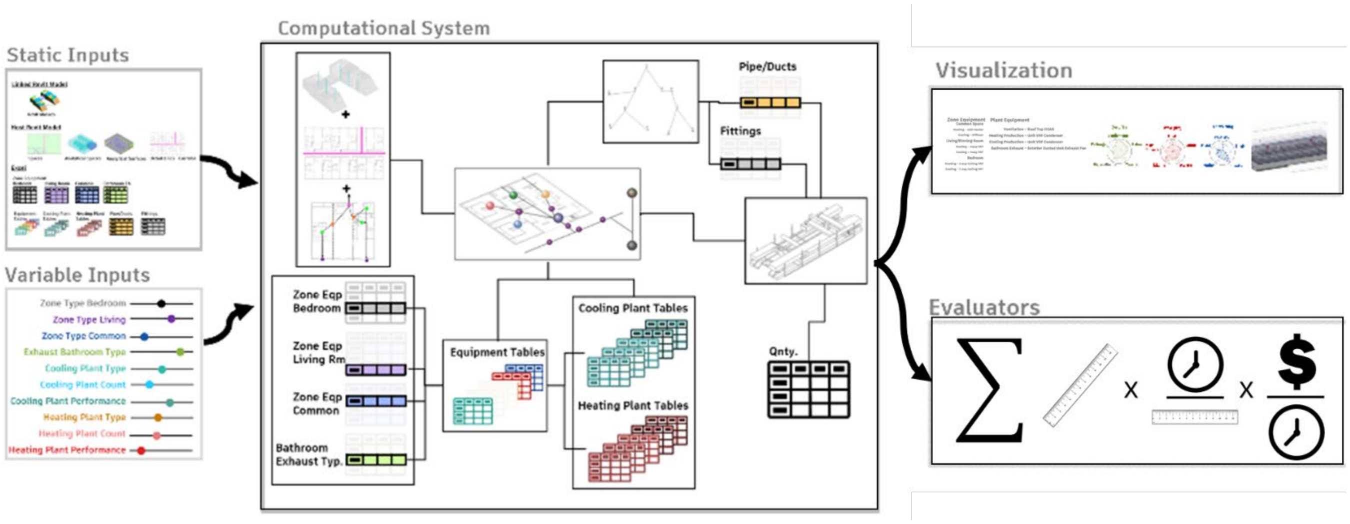


Another critical data source is a procurement database containing a library of all the materials used to construct an array of HVAC systems. This manufacturing-specific data is imperative for the generative design study to output genuinely accurate results, aka garbage in garbage out. Performing the procurement process before the design phase provides material costs, equipment dimensions, engineering specifications, and other critical information. Downstream automation also benefits. For example, manufacturing specifications can be used to instantly populate engineering equipment schedules. Without these data sources, the generative design workflow outlined below would not be possible. It is also important to note that this workflow is always an ongoing project focusing on continuous improvement. More equipment options can be easily added, and data points like costs can easily be updated.



Problem Formation

iBuilt sought a standardized HVAC system that was cheap, easy to install, small in size, and out of sight. This study explores different HVAC systems and layout configurations while analyzing and illustrating the trade-offs between the chosen objectives. The computational system integrates various constraints, including routes for ducts/pipes, heating and cooling zone locations with fixed thermal loads, and required ventilation rates for units and shared spaces. The primary algorithm used by the computational system is a graph that represents the connections between HVAC equipment. In mathematics, graphs are a way to represent a network of connected objects. The relationship between the objects called vertices is done with edges. A graph represents an HVAC system, edges represent pipes/ ducts, and vertices represent mechanical equipment, fittings, or accessories. Utilizing this data structure the flows (CFM / GPM) is calculated through the system, enabling the pipe and duct sizing and ultimately generating the required data needed to calculate the objective functions. The workflow generates different design options by cycling through variables; changing equipment types, quantities, and locations. Below is the high-level diagram of the workflow.



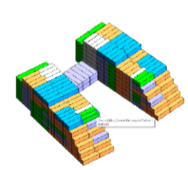
Static Inputs

To minimize the time required to run a generative design study, it is best to perform as many calculations outside of the study as possible. For example, heating and cooling loads are calculated beforehand since they are fixed during the system selection process. The beauty here is the interoperability with Revit's database and design tools. This workflow takes advantage of this and gathers data from a few different sources. Revit masses are collected from a linked model. Spaces, analytical spaces, and analytical surfaces are used from the host model. Procurement data (equipment tables, fittings, pipes/duct, and system definitions) are stored in an Excel file. The element parameters and geometry from Revit and data tables from

Revit Data Dictionaries

Revit masses

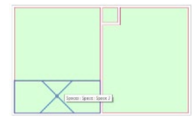
unit type	An id to identify unique apartment layouts.
unit id	The unit number the module belongs to.
solid	The solid block that represents a module
level	The Revit level name of the mass



Excel are organized into Dynamo dictionaries and stored in the script using the **Data.Remember** node. Below is a description and the content of these dictionaries:

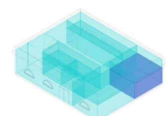
Revit Spaces

unit id	The unit number the space belongs to.
space type	The classification for ventilation and exhaust rates.
minimum exhaust	The required exhaust rate (CFM)
solid	The 3D solid of the Spaces interior.
point	The space's location point.
surface bottom	The space's floor surface.
surface top	The space's floor surface.



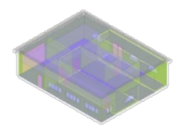
Revit Analytical Spaces

space id	The id of the Revit Spaces the analytical space was created from.
peak cool	The peak cooling load calculated by the energy analysis. (btu/hr.)
peak heating	The peak heating calculated by the energy analysis. (btu/hr.)



Revit Analytical Surfaces

id	The modular id that the surface bounds.
surface	The exterior and roof surfaces from the energy model.



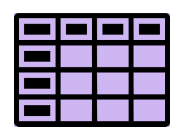
Revit Detail Lines

line	Lines running along the center of the corridors.
level	The Revit level name the line is on.

Excel Data Dictionaries

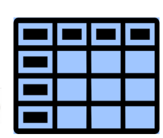
Pipes/Ducts

material	The material (copper)
type	The material type description (Type L)
maximum flow	The maximum allowable flow rate (gpm) or (cfm)
maximum pressure	The maximum allowable pressure (psi)
size	The nominal diameter or length and width (in) or (in x in)
weight	The weight of the material per foot (lb./ft)
price	The coast of materials per foot ($/ft)
labor	The installation time (min)



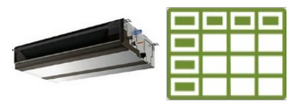
Fittings

material	The material (copper)
type	The material type description (Type L)
maximum flow	The maximum allowable flow rate (gpm) or (cfm)
Fitting Type	The type of fitting (Tee)
maximum pressure	The maximum allowable pressure (psi)
size	The sizes of each connector (in x in x in)
connectivity	The connection method used (Solder)
weight	The weight of the material per foot (lb./ft)
price	The coast for each ($)
labor	The installation time (min)



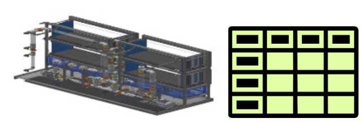
Zone Equipment

equipment type	The description of the equipment (unit ventilator)
cooling coil	The type of cooling coil (chilled water)
cooling capacity	The maximum amount of heat the units can move (btu/hr)
heating coil	The type of cooling coil (gas)
heating capacity	The maximum amount of heat the units can provide (btu/hr)
manufacturer	The maker of the equipment (ACiQ)
model	The model id of the equipment (ACiQ90)
mounting type	Where the equipment is placed (wall)
height	The height (in)
length	The length (in)
width	The width(in)
weight	The weight (lbs.)
revit family	The associated name of the Revit Family.
revit type	The associated name of the Revit Family Type.
revit analytical system	The associated number for the Revit Analytical System.



Central Plants

equipment type	The description of the equipment (unit ventilator)
cooling coil	The type of cooling coil (chilled water)
cooling capacity	The maximum amount of heat the units can move (btu/hr)
heating coil	The type of cooling coil (gas)
heating capacity	The maximum amount of heat the units can provide (btu/hr)
manufacturer	The maker of the equipment (ACiQ)
model	The model id of the equipment (ACiQ90)
mounting type	Where the equipment is placed (wall)
height	The height (in)
length	The length (in)
width	The width(in)
weight	The weight (lbs.)
revit family	The associated name of the Revit Family.
revit type	The associated name of the Revit Family Type.
revit analytical system	The associated number for the Revit Analytical System.



Variable Inputs

For this study, countless different variables can drive variations in the computational systems. Some examples include switching equipment types, changing fitting types, or varying the number of central plants. Since the sliders minimum and maximum values need to be determined upfront, the tricky part was constraining these variables so that the sliders work for every situation. For instance, if a slider were used to alternate the type of central plant, then there would need to be the same number of central plant options for a water system and an air system.

Slider Settings

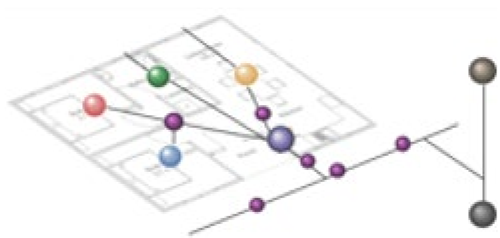
Variable Name	Min	Max	Description
index room zone	1	# Options	The row index of the single space zone equipment.
index living zone	1	# Options	The row index of the living room zone equipment.
index common zone	1	# Options	The row index of the common zone equipment.
index ventilation	1	# Options	The row index of the ventilation strategy.
index cooling plant	1	# Options	The row index of the cooling plant type.
index heating plant	1	# Options	The row index of the heating plant type.
# bathroom exhaust fans	0	10	The row index of the bathroom equipment. (0 = decentralized)
# supply fans	0	10	The row index of the bathroom equipment. (0 = decentralized)
# Cooling plants	0	10	The number of cooling plants (0 = decentralized)
# Cooling plants	0	10	The number of heating plants (0 = decentralized)
unit mech position	0	1	The location of the apartment units' mechanical closet
floors per plant	0	5	The maximum number of floors per plant.

Computational System

With the assortment of variable inputs, the computational system created in the Dynamo graph can generate thousands of different HVAC system configurations, effectively covering all possible solutions. The computational system can be broken down into several sub algorithms that are described below:

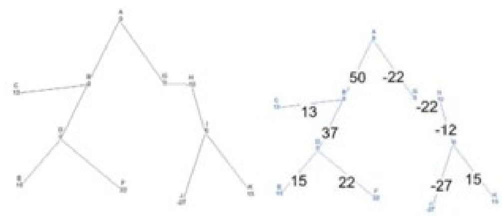
The first step uses the equipment index variables to select different type of mechanical equipment configurations. The equipment locations and quantities are also calculated with variables.

The next step generates the graph by combining the corridor lines, mechanical room location and the shafts.

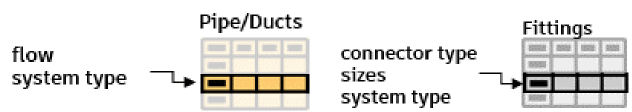


Once the graph is formed, the nodes are assigned a piece of mechanical equipment or fitting and the edges are assigned a pipe of duct material.

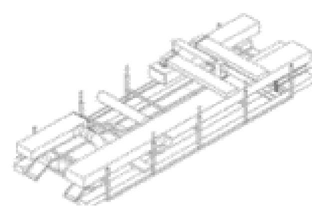
The next step is to calculate the flow for the pipe and ducts. This is done by converting the graph into trees for all the different systems. A tree is a simple graph structure defined by a set of rules: one root node may or may not connect to others, but ultimately, all connections stem from one specific place. Using this structure, calculating flow (gpm,cfm) back to a root node is a relatively straightforward process. With the flows calculated, the sizes can be determined.



Now with the flow and system types known for the ducts and pipes, the sizes, prices, labor, etc. can be looked up from the procurement data tables. The same process is done with the fitting but use the ducts and pipes sizes as inputs.



At this stage, the sub-assemblies can be configured by packing all the ducts and pipes together.



We then move on to extracting the needed data to calculate the objective function.

Objective Functions

The high-level objective metrics used to measure iBuilt's goals are listed in the table below.

Name	Description
Material Cost	Capital cost of all construction materials.
Labor Cost	Time required to install multiplied by the labor cost.
Shaft Area	Total area needed to fit the infrastructure into vertical shafts.
Max Main Area	Maximum area needed to fit the infrastructure into horizontal mains.
Installation Factory	Labor time required to install infrastructure on the factory floor.
Installation Site	Labor time required to connect and commissioning HVAC system on site.
Weight	Total weight of the HVAC equipment.
Exterior Wall Area	Total area of outside wall penetrations from lovers and equipment.

Each performance function is the summation of measurable values that can either be maximized or minimized. For example, the overall cost is the sum of material and labor costs.

$$\mathbf{Cost_{material} = Duct_{length}\,(Duct_{\$/ft}) + Pipe_{length}\,(Pipe_{\$/ft}) + Cost_{equipment}}$$

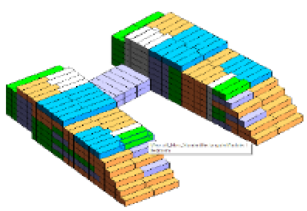
$$\mathbf{Cost_{labor} = Duct_{length}\,(labor_{\$/ft}) + Pipe_{length}\,(labor_{\$/ft}) + (fitting\ qnt.)Cost_{labor}}$$

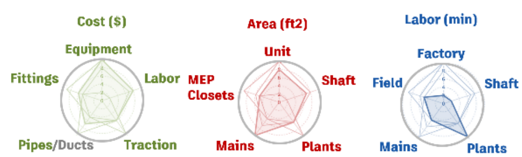
$$\mathbf{Cost_{capital} = Cost_{labor} + Cost_{material}}$$

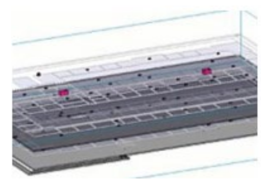
Both material and labor costs are calculated by adding material lengths and quantities times cost.
Note more granular objectives could calculated for further analysis. For example, the weight of roof top equipment could be used to incorporate crane sizes and pricing. Once the overall system is developed it is easy to add more data and add more measurable objectives.

Visualization

Three graphics are created in Dynamo to help convey information about the results. First, the text displays the system and equipment types. Second, a series of spider graphs shows the breakdown of the main objectives (cost, labor, areas, visibility). Third, a color-coded 3D model of the infrastructure routing.

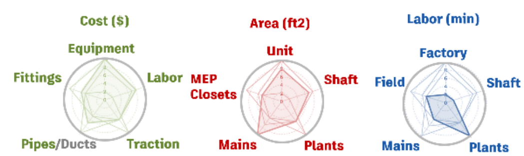




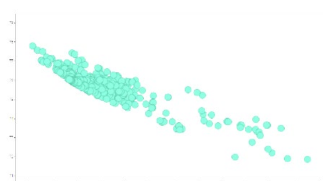


Results

The result is not an answer; instead, a set of tools for conducting an insightful and comprehensive analysis of all the possible designs. A single optimization can be performed on the overall cost function. The optimized method is selected for this study, and the only goal checked is the "overall cost." The resulting spider graphs and 3D thumbnail make it clear that the cheapest system needs exterior wall penetration.

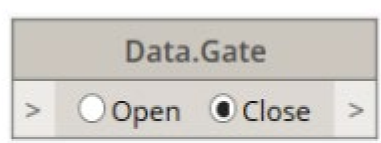


A multi-objective approach allows the exploration of the design space. Using the scatter plot Pareto optimal solution can be found between objectives.



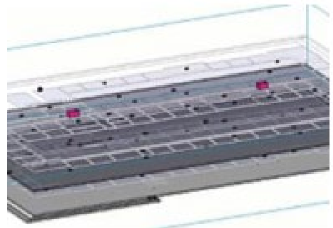
Revit Integration

Once the solution is selected, the results can be integrated into the Revit project, saving time and ensuring a quality foundation for the rest of the design, manufacturing, construction, and operations. Like the **Data.Remember** node the **Data.Gate** node controls the flow of data. The difference being the node belongs downstream of the script's generative section rather than before. After the **Data.Gate** node, Revit API action can once again be called, allowing the creation of new Revit elements. Connecting into the **Data.Gate** node is the output dictionary exported generated by the study. The dictionary contains all the information needed to integrate the results into the Revit project. Below are some examples of different Revit elements created from the HVAC System selection study.



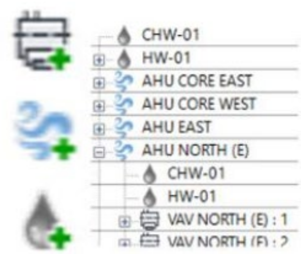
Revit MEP System

Exported from the study is the output dictionary that contains all the information needed to populate the Revit Families that make up the system and fill out all the engineering schedules.



Systems Analysis

Analytical Zone Equipment, Water Loop and Heating loops can be added to the Revit model. This function allows energy modeling straight from the Revit model. It attaches Zone Equipment and Central Plant equipment like boilers, chillers, and air handlers from an extensive selection of objects and relationships. From this, we can quickly test all our variations with Energy Plus — the industry's leading building energy systems simulation engine and Open Studio SDK. By integrating these automation methods into the generative design study, modeling takes a fraction of the time while guaranteeing a standardized and quality start to the rest of the project.



Creating a generative design workflow for iBuilt's HVAC system selection process proved to be an excellent approach to this enormous challenge. The workflow provided the systematic analysis expected from the CEO, supplied a framework to guide the formulation of the problem, and supplied transparency between inputs to outputs. The outcome of this study provided a significantly slimmed-down range of design options that only contained optimal configurations. These potential options can be narrowed further based on live data and desired outcomes. Also, the workflow supplied a foundation for further automation by integrating any of the chosen designs into the project.

Conclusion

This work demonstrates the potential of generative design. Digital systems like this open a whole new realm of possibilities in the engineers' tool kit. Leveraging these tools not only leads to better design but also offers immeasurable other benefits. For instance, establishing time-saving automation, forcing data organization, optimizing for targeted objectives, and downstream automation becomes a breeze. With this transparency and dynamic control, team discussion about the design become energetic where objectives and tradeoffs can be explored on the fly, assumptions can be tested, and compromise can be debated with data.

HVAC system selection is just one of the countless multi-objective problems found in various building designs. Designers must provide strict standardization and creative thinking to translate their knowledge into mathematical language, notation, and rule-based geometric systems with a set of measurable goals. The generative design framework provides a fairly simplified path for solving these challenging problems. Complex mathematical programming is replaced with easy-to-comprehend virtual programming and data-already-in-project models.

This emerging movement with its newfound ideas has been assigned the name "generative design" for better or worse. Regardless of the terms, the underlying philosophies will inevitably have an impact on the AEC industry. These processes automate historically labor-intensive tasks and let designers shift focus, delivering better engineering services. The efficiency achieved directly translates to time and cost savings for the business. Because of these potential savings, I foresee the future of building design being system optimization. As the technologies continue to improve, coding becomes more prevalent, more design knowledge gets digitally automated, and data centric workflows will inevitably become common practice. No longer will projects start from a blank page. Instead, architectural, structural, and engineering computational systems will merge into a holistic multi-objective.