SIGRADI2018
TECHNOPOLITICAS
xxii congresso da sociedade iberoamericana de gráfica digital
22th conference of the iberoamerican society of digital graphics
07|08|09|novembro|2018
iau usp | são carlos | sp br

Generative design in the design development of metallic constructions

Renato Godoi da Cruz
Universidade Federal de Ouro Preto | Brazil | renatogcruz@hotmail.com

Cláudia Maria Arcipreste
Universidade Federal de Ouro Preto | Brazil | claudiaarcipreste@gmail.com

Rafael Lemieszek Pinheiro
Faculdade Pitágoras | Brazil | lemieszek@gmail.com

Rovadávia Aline de Jesus Ribas
Universidade Federal de Ouro Preto | Brazil | roviaaline@gmail.com

Abstract

The present article describes the construction of a system that combines parametric modeling strategies and genetic algorithms for optimization. By means of the reformulation of the Darwinian evolutionary process, it is sought to systematize a project process that allows the architect to act in the parameterization of the problems, beyond the mere formal proposition of solutions, in favor of the exploration of a greater variety of projective possibilities than would be possible using traditional design methods.

Keywords: Generative design; Evolutionary algorithms; Structural analysis; Environmental analysis and Metallic construction.

INTRODUÇÃO

Este artigo tem como objetivo refletir sobre os limites do pensamento científico na prática projetual, cujas bases teóricas têm repousado no modelo racional da determinação técnica e cientifica, e avaliar o estado da arte da prática de projeto em arquitetura sob a perspectiva de um pensamento arquitetônico mais polivalente.

Aplicando critérios de desempenho e de otimização estrutural, a intenção é sistematizar campos de conhecimento comuns da arquitetura, da engenharia e de outras disciplinas – matemática e ciências da computação, numa articulação transdisciplinar dentro de um cenário próprio, mas configurável e disposto a incluir outros saberes. Para isso, combinamos estratégias de modelagem paramétrica e de algoritmo genético para otimização no desenvolvimento de uma estrutura em aço para a cobertura da arquibancada do campo de futebol de uma instituição de ensino.

O processo tradicional de projeto pode ser descrito como um processo consciente de experimentação: hipóteses de solução formal são formuladas e testadas (em desenhos e modelos) até que o projetista se dê por satisfeito com o resultado. Neste trabalho, propomos a utilização de ferrmamentas paramétricas para explorar o máximo número de hipóteses de projeto dentro de um conjunto de variáveis estipuladas *a priori* com o objetivo de explorar complexidades que manualmente seriam, em termos práticos, impossíveis de serem testadas.

Para a modelagem geométrica, utilizou-se um *software* de interface em *Visual Programming Language* (VPL) que opera a partir de um editor baseado em nós que conduz a uma serie de operações geométricas para produzir uma geometria flexível, baseada em parâmetros de entrada. Esses parâmetros, por sua vez, são controlados por um algoritmo genético para otimizar o modelo a partir de duas metas: (1) minimizar o peso e (2) maximizar o potencial energético. Juntas, essas duas estratégias criam uma abordagem que garante alto desempenho, ao mesmo tempo em que permite que o arquiteto atue orientado não pela forma, mas pelo conjunto de condicionantes (físicas, ambientais, locais) que resultará, dialogicamente, na forma

Para estruturar e conduzir o trabalho, foram definidos como objetivos específicos: (1) a revisão do conteúdo teórico e prático da arquitetura baseada na prática projetual onde o arquiteto abre mão do controle da definição especifica da solução em favor da exploração de uma variedade maior de projetos do que seria possível usando métodos de *design* tradicionais; (2) a aplicação deste conteúdo em projeto arquitetônico de construção metálica, bem como a descrição deste processo como forma de contribuição como base teórica de suporte para futuros desenvolvimentos por parte de arquitetos e outros profissionais envolvidos em processos de design.

REVISÃO DE LITERATURA

O processo de projeto tende a ser altamente dependente do caminho percorrido pelo arquiteto. Isso significa que as decisões iniciais determinam e limitam as

possibilidades do projeto em estágios posteriores. À medida que o processo de processo avança no tempo, as decisões são tipicamente tomadas quando são confrontadas. A consequência deste fato é que o projeto acaba sendo o resultado de aproximações sucessivas, em pequenos ciclos de análise, síntese e avaliação (Lawson, 1983). A competência do arquiteto para isso está em sua habilidade de retomar, reestruturar, adaptar e aplicar ideias relevantes advindas de projetos anteriores (Oxman & Oxman, 1992).

Entretanto, isso pode se tornar um problema, pois quando os conceitos dos arquitetos tornam-se um padrão de pensamento, as soluções para novos problemas podem se restringir na mera reprodução de soluções anteriores, colocando em segundo lugar a descoberta de novas possibilidades e encaminhamentos para novas soluções (Buchanan, 1992).

Além disso, a complexidade do espaço construído e a subjetividade significam que, em Arquitetura (projeto), não há uma única solução (resposta) para um dado problema (pergunta). A partir desse ponto, o arquiteto Caio Adorno Vassão termina seu livro *Metadesign: ferramentas, estratégias e ética para a complexidade* (2010) propondo o projeto como "pergunta". Essa ideia contradiz o entendimento linear do projeto como uma resposta natural a uma pergunta.

Abordagens de ordem não-teoremática do projeto não são algo novo, com diferentes profissionais de projeto adotando metodologias projetuais que abrem oportunidades para o surgimento de resultados dos quais são estruturalmente diferentes dos obtidos pelos procedimentos convencionais, se não em seus produtos finais, ao menos nos caminhos para se chegar até eles. Um exemplo é Antoni Gaudí e seus experimentos com redes suspensas (Figura 1a). Presas na face inferior de uma superfície plana, a estabilidade destas redes resulta em representações invertidas de catenárias. Esse processo fui utilizado na definição de um conjunto de catenárias para o projeto da *Sagrada Família*, construída na cidade de Barcelona, Espanha. Outro exemplo é o estudo *Optimized Path Systems*, de Frei Otto, do início dos anos 1990. Similar à técnica utilizada por Gaudí na *Sagrada Família*, o método analógico conhecido como máquina de fios de lã foi usado para gerar modelos de cidades e seus sistemas de circulação, simulando fisicamente as forças (Figura 1b).

Análogo ao sistema de Frei Otto é o processo de concepção do projeto *Oblique WTC* (Figura 1c), de Lars Spuybroek. Semelhante ao modelo de fios de lã de Otto, a técnica de fios de lã de Spuybroek consiste em duas estruturas quadradas com marcações de *grids* ortogonais e conectadas por fios de lã. Quando esse conjunto é mergulhado em água e balançado horizontalmente, os fios imediatamente começam a se auto-organizarem em um sistema complexo de ramificações. Após este processo, o modelo foi digitalizado e cada fio ganhou mais espessura constituindo os núcleos das torres que compõem o complexo (Ferrante, 2013).

Estes exemplos da reinterpretação e da ressignificação da cultura de projeto possuem duas características importantes: (1) são processos fluídos, com resultados continuamente ajustados a partir da alteração das condicionantes; (2) são exemplos de processos focados na produção de processos (perguntas) ao invés do produto (resposta).

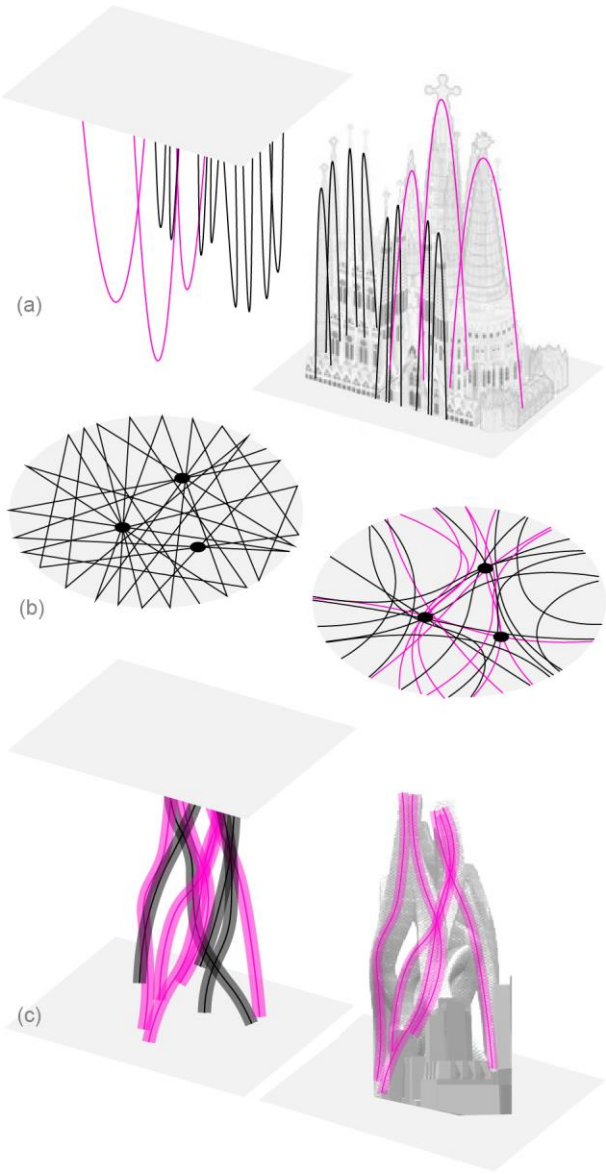


Figura 1: Abordagens não-teoremática de projetos. Fonte: autores.

A primeira característica se refere a uma aplicação customizada de procedimentos científicos (modelagem) para a síntese da forma. Estas metodologias de projeto se contrapõem à tentativa de delimitar uma base científica mais geral (teorema), criticada anteriormente por Vassão. A segunda característica diz respeito ao controle ou, mais especificamente, a falta dele no processo de projeto. Em todos os três casos, os arquitetos perderam controle das definições específicas das soluções para operarem na formalização dos problemas.

O problema do controle em arquitetura, descrito acima, pode ser compreendido como a qualidade do modelo que o arquiteto emprega para gerencia-lo. Segundo a Lei da Variedade Requerida de William R. Ashby (1970), para que um sistema seja regulador de outro sistema, exige-se

deste que se disponha de pelo menos tantos estados (variáveis) quanto o sistema que se deseja controlar possa apresentar. Caso contrário, o que temos é um controle arbitrário que, segundo Glanville (2002) acaba assumindo a forma de restrição. No contexto do projeto em arquitetura, podemos ver esse tipo de gerenciamento no processo de projeto tradicional, compreendido como solução (resposta), que acaba condicionando o projeto a soluções formais presentes no imaginário do arquiteto a partir de experiências anteriores.

A aplicação de soluções de design originais e mais adequadas à situação presente é, assim, frequentemente preterida em favor da mera aplicação de vocabulários formais previamente existentes.

Assumir estar fora do controle na prática de projeto não é aceitar um *design* menor ou incompleto, mas entendê-lo como uma oportunidade de alteridade e de mutabilidade que estão à disposição de uma operação de projeto. Essa compreensão não deve configurar uma transformação radical frente às produções cotidianas recorrentes, mas pode se incorporar às teorias desenvolvidas dentro de seus próprios campos e outras que viabilizem a ampliação de seus limites.

Para Nagy (2017) uma boa estratégia para isso está no modo como o *design* é projetado na natureza. Para o autor, os organismos encontrados na natureza apresentam grande diversidade de soluções formais que vão muito além da imaginação humana. Ainda para o autor, mesmo que ainda não possuamos as ferramentas e o conhecimento necessário para projetar sistemas totalmente naturais, conceitos matemáticos nos fornecem uma maneira concreta de reformular o processo evolutivo como fundamentalmente um processo de otimização.

Segundo Yang (2010), na Matemática, é possível escrever a maioria dos problemas de otimização na forma genérica:

$$\underset{x \in R^n}{minimizar}\ f_i(x), (i = 1, 2, \dots, M),$$

$$sujeito\ a\ h_i(x) = 0, (j = 1, 2, \dots, J),$$

$$g_k(x) \leq 0, (k = 1, 2, \dots, K),$$

sendo $f_i(x)$, $h_i(x)$ e $g_k(x)$ funções do vetor de *design*:

$$x = (x_1, x_2, \dots, x_n)^T$$

Em que os componentes x_i e x são chamados de variáveis de projeto. As funções $f_i\ (x)$, onde $i = 1, 2, \dots, M$, são chamadas de funções objetivas. Podemos classificar a otimização como mono-objetiva, quando $M = 1$, ou multiobjetivo, quando $M > 1$. O espaço atingido pelas variáveis de decisão é comumente chamado de espaço de projeto R^n, enquanto o espaço formado pelos valores da função objetivo é chamado de espaço de solução e as igualdades h_j e desigualdades g_k são chamadas de restrições (Yang, 2010).

Problemas de otimização podem ser operados por diversas abordagens, sendo comum a utilização de algoritmos. De maneira sintética, podemos classifica-las em duas categorias básicas: (1) métodos determinísticos – aplicação direta de uma série de etapas definidas; (2) métodos estocásticos – introdução de algum nível de aleatoriedade enquanto buscam uma solução.

O método escolhido depende da maneira como o problema de otimização é descrito e do tipo de informação que se dispõe sobre as funções de objetivo e restrição. Para este trabalho, o algoritmo para otimização de projeto baseado no método estocástico conhecido como Algoritmo Genético (GA) é particularmente interessante porque suas regras e operações são inspiradas na evolução darwiniana e na seleção natural de sistemas encontrados na natureza (Shiffman, 2012), o que nos permite explorar o processo de projeto em arquitetura por meio do uso de algumas das potencialidades do *design* natural.

Em Ciência da Computação, algoritmo genético se refere à técnica de buscar um resultado específico para um dado problema. Muito do que pensamos hoje como algoritmo genético foi desenvolvido por John Holland, cujo livro *Adaptation in Natural and Artificial Systems* (1975) foi pioneiro na pesquisa deste campo de estudo, apresentando os fundamentos teóricos e explorando aplicações desde agentes da teoria econômica até projeto de dispositivos complexos como turbinas de aeronaves e circuitos integrados. Hoje, os algoritmos genéticos fazem parte de um campo mais amplo de pesquisa, frequentemente chamado de Computação Evolucionária que compreende o ramo da inteligência computacional e da computação natural (Shiffman, 2012).

No contexto da arquitetura, um dos métodos que se utiliza desse tipo de algoritmo é chamado de *design* generativo. Sua finalidade é explorar, de maneira ágil e eficaz, novas possibilidades do projeto, dentro de um universo de possibilidades imensuráveis. O *design* gerativo pode ser mais amplamente definido como o emprego de um sistema gerador (um conjunto de regras, um programa de computador, um conjunto de transformações geométricas, um diagrama etc.) no processo de *design* por meio do qual o projeto final emerge.

METODOLOGIA

Os procedimentos metodológicos aplicados neste trabalho foram divididos em quatro etapas: (1) a primeira parte consiste na pesquisa teórica referencial dedutiva para assimilar o projeto como processo; (2) em uma segundo etapa foi aplicado o processo evolutivo em um estudo de caso; (3) na terceira etapa realizou-se a análise quantitativa dos dados obtidos, aplicando-se gráficos de dispersão e a seleção dos melhores indivíduos (*designs*); (4) em uma quarta etapa as discussões e ponderações foram organizadas e descritas.

A segunda etapa se desdobrou em outras três partes: (2a) a construção do espaço de projeto, ou seja, o sistema que gera todas as soluções possíveis para um determinado problema de projeto. (2b) o desenvolvimento de medidas para julgar o desempenho de cada projeto, primeiramente abordado com apenas uma função objetiva, e a minimização do peso total e em seguida, acrescentando-se um segundo objetivo, a maximização do potencial fotovoltaico, que somado ao objetivo

anterior, o transforma em um problema multiobjetivo; (2c) a aplicação de algoritmos evolutivos para pesquisar o espaço de projeto e encontrar projetos de alto desempenho. Para isso, foi utilizado o *software Rhinoceros*, para modelagem tridimensional e seu *plug-in* de modelagem paramétrica *Grasshopper*. Também foram aplicados os seguintes *add-ons* para *Grasshopper*: (1) *Karamba*, para análise estrutural; (2) *Ladybug*, para análise ambiental e (3) *Discover*, um *framework* para projeto generativo.

FERRAMENTA E ESTRATÉGIA PARA O *DESIGN* GENERATIVO

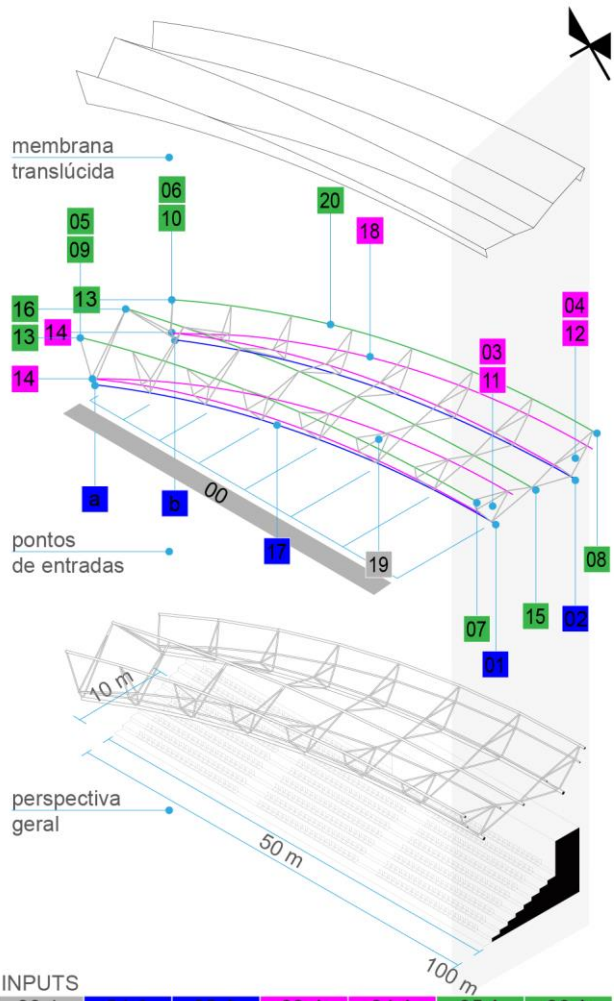
O conceito de *design* generativo, como descrito anteriormente, atribui ao algoritmo a função de explorar o espaço de projeto de forma semiautônoma. Tradicionalmente, tal abordagem tem sido usada para gerar modelos de alto desempenho, com base em objetivos predefinidos. Para isso, seguiremos as recomendações de Villaggi *et al.* (2018), que dizem que, para se construir um modelo generativo, deve-se definir: (1) um modelo de geometria flexível que define um espaço de projeto; (2) um conjunto de medidas que descreva os objetivos e as restrições de projeto e (3) um algoritmo genético que possa buscar no espaço de projeto opções de alto desempenho com base nos objetivos estabelecidos.

CONSTRUÇÃO DO ESPAÇO DE PROJETO

Na arquitetura, modelos de geometrias flexíveis podem ser especialmente construídos por meio de *softwares* de modelagem paramétrica. Um dos mais conhecidos, o *Grasshopper*, usa operações matemáticas e geométricas definidas por um conjunto de parâmetros capaz de conduzir uma serie possibilidades formais antes de conduzir a uma forma final.

A modelagem paramétrica, que permite a construção de modelos flexíveis, também possibilita obter resultados de grande complexidade a partir de princípios simples, pois amplia a definição de projeto-indivíduo – *design* estático, portanto não flexível – para o projeto-sistema – aquele que codifica uma espécie inteira – ou seja, aquele que projeta um espaço de projeto, dentro da concepção Matemática dos problemas de otimização descrita anteriormente por Yang (2010).

Seguindo os princípios da arquitetura paramétrica, que se refere à automação de parâmetros, iniciou-se a construção do modelo, uma cobertura treliçada com cobertura em membrana translucida, a partir da determinação espacial de um conjunto de parábolas, subdivididas e interligadas por diagonais, conforme a Figura 2. As duas primeiras parábolas, formadas pelas barras A, são controladas por parâmetros de entrada no ponto médio (*eixo z*). Como formam a base da estrutura, estas parábolas são apoiadas em suas extremidades. Em seguida, outras quatro parábolas, formadas pelas barras B, são controladas por parâmetros de entrada no ponto médio (*eixos z* e *y*) e nas suas extremidades (*eixo x* e *z*). As três últimas parábolas, formadas pelas barras C, são controladas por parâmetros de entrada no ponto médio (*eixo z*) e nas extremidades (*eixos x e z*). Esse conjunto de parábolas, por sua vez, é controlado por um parâmetro de entrada (divisão) e interligados em seus pontos por diagonais (barras D). Por fim, cada conjunto de barras é controlado por um parâmetro de entrada que permite definir um perfil diferente para cada tipo de barra (A, B, C e D), independentemente, a partir de uma tabela com as 30 seções tubulares circulares mais frequentemente produzidas pela empresa *Vallourec*.



INPUTS

00 /	01 ↑	02 ↑	03 ↑	04 ↑	05 ↑	06 ↑
10	5.00	5.00	1.00	1.00	0.00	0.00
20	10.0	10.0	4.00	4.00	1.00	1.00
07 ↑	08 ↑	09 ↑	10 ↑	11 ⊿	12 ⊿	13 →
2.00	2.00	2.00	2.00	10	10	0.00
5.00	5.00	7.00	7.00	20	20	1.50
14 →	15 ↑	16 ↑	17 ○	18 ○	19 ○	20 ○
0.00	1.00	4.00	0	0	0	0
1.50	5.00	10.0	29	29	29	29

APOIOS	rotação			translação		
a / b	x ✓	y ✓	z ✓	x ✗	y ✗	z ✗

LEGENDAS

barras A	barras B	barras C	barras D	divisor	eixo X	eixo Y

simetria	perfil	ângulo	livre	impedido

Figura 2: Diagrama do modelo e seus parâmetros de entrada. Fonte: autores.

Cada um dos parâmetros de entrada descritos na Figura 2 controla independentemente uma característica específica, e a relação entre todas as variáveis resultará em uma proposta formal única. Em analogia com a evolução natural, o conjunto de parâmetros que descreve o sistema representa o genótipo, conjunto dos

cromossomos que determinam todas as possibilidades formais de um organismo, enquanto o projeto final representa o fenótipo, resultado da expressão dos genes do organismo.

O espaço de projeto deste modelo possui 21 entradas com valores que variam de 10 a 500 caracteres. Ainda que a complexidade possível não seja imediatamente aparente, o total de soluções possíveis deste espaço de projeto é de 2,21 x 10^{42}. O teste manual de cada uma das hipóteses é virtualmente impossível, uma vez que, considerando o gasto hipotético médio de 10 minutos por hipótese, seria necessário dedicar 1,76 x 10^{25} vezes a idade do planeta Terra, considerando sua idade estimada em 4,54 x 10^{9} anos (Shiffman, 2012). Para se viabilizar, o processo tradicional de projeto descarta virtualmente todas as soluções possíveis em favor das poucas que se alinhem com o conhecimento prévio e os gostos pessoais do projetista.

AVALIAÇÃO DO MODELO

Para aplicar o algoritmo genético visando evoluir projetos de alto desempenho, o modelo deve também conter um conjunto de medidas que informam ao algoritmo quais projetos têm melhores desempenhos. Neste trabalho, o modelo usa análise de elementos finitos (FEA) para simular o desempenho de cada *design* sob as condições de dimensões e de cargas dadas. Essa análise nos fornece um conjunto de métricas que podemos usar para estabelecer os objetivos do nosso problema de otimização: (1) peso total do projeto (objetivo a minimizar); (2) potencial energético da cobertura (objetivo a maximizar).

Devido à natureza do nosso experimento, a análise estrutural é construída como etapa fundamental na elaboração do problema de otimização e usada como restrição. Para se obter a resposta da estrutura perante as ações que são aplicadas ao modelo, é utilizado o *Karamba*, *add-on* que calcula os deslocamentos e os esforços solicitantes nos elementos de modelo construídos no ambiente *Grasshopper*.

Como os resultados obtidos na análise estrutural influenciam diretamente na viabilidade do modelo, os critérios adotados como condição para o projeto são os seguintes: (1) dimensionamento dos estados limites das barras submetidas à força axial de compressão, onde a força axial de compressão solicitante (Nc,sd) deve ser menor ou igual à força axial de compressão resistente de cálculo (Nc,rd); (2) dimensionamento dos estados limites das barras submetidas à força axial de tração, onde a força axial de tração solicitante (Nt,sd) deve ser menor ou igual à força axial de compressão resistente de cálculo (Nt,rd); (3) dimensionamento dos momentos fletores, onde o momento fletor solicitante (Msd) deve ser menor ou igual ao momento fletor resistente (Mrd); (4) dimensionamento do deslocamento máximo, onde a flecha máxima ($Max.Displacement$) deve ser inferior ao quociente do comprimento do maior vão pelo número trezentos ($l/300$).

A abordagem adotada para lidar com a restrição neste trabalho é de caráter punitiva, ou seja, os projetos que quebrarem a regra de restrição são imediatamente punidos, desqualificados, o que os impedem de participar dos próximos cruzamentos (Yeniay, 2005). O método de penalidade neste caso ocorre de duas maneiras. Primeiro na forma multiplicativa:

$$eval(x) = \begin{cases} f(x) \\ f(x)p(x), \end{cases} \qquad if\ x\ \in F$$

Sob essa convenção, a função objetivo é determinada por $eval\,(x)$, onde $p\,(x)$ representa um termo de penalidade na lógica de verificação estrutural. Se nenhuma violação ocorrer, $p\,(x)$ é tomado igual à zero. Caso contrário, desqualifica-se o projeto atribuindo $p\,(x)$ igual a dez mil (10.000). Essa lógica é aplicada na avaliação das análises de compressão ($Nc,rd \geq Nc,sd$), tração ($Nt,rd \geq Nt,sd$) e do momento fletor ($Mrd \geq Msd$).

A segunda maneira ocorre na forma aditiva. Esta convenção é aplicada exclusivamente no critério de deslocamento ($Max.Displacement \leq l/300$):

$$eval(x) = \begin{cases} f(x) \\ f(x) + p(x), \end{cases} \qquad if\ x\ \in F$$

Este conjunto de métricas (dois objetivos e uma restrição) permite que um algoritmo procure automaticamente por projetos de alto desempenho. O processo punitivo também permite ao sistema identificar, nos desenhos desqualificados, em que sentido as variáveis devem caminhar para resultar em um desenho qualificado.

DEFINIÇÃO DO ALGORITMO GENERATIVO

Para uma busca efetiva das opções de alto desempenho, precisamos de um sistema externo que possa trabalhar juntamente com o modelo. Uma das opções seria um algoritmo de busca. A vantagem desse tipo de algoritmo é que ele não precisa receber informações prévias sobre o funcionamento interno do modelo. Isso porque a estratégia geral desses métodos é iniciar uma amostragem aleatória e, em seguida, usar o conhecimento derivado dessa amostra para obter outras amostras com melhores desempenhos. Seguindo esse processo, o algoritmo busca as melhores opções sem saber nada sobre como o espaço de projeto funciona e sem ter que testar todos os projetos possíveis.

Os algoritmos meta-heurísticos são particularmente interessantes porque suas regras e operações são inspiradas nos processos evolutivos da natureza, o que nos permite explorar o *design* de maneira semelhante ao que ocorre no *design* natural.

O significado de *meta-heurísticas* relaciona-se ao conjunto de regras que orientam este processo. Nesse caso, as regras são *meta* porque são aplicadas a um algoritmo geral, e não a um problema específico. Isso significa que as abordagens meta-heurísticas podem ser generalizadas a qualquer problema de otimização.

O *Discover*, de Danil Nagy, é uma estrutura flexível e modular para projeto generativo e exploração do espaço de projeto baseado neste tipo de algoritmo. Especialmente adaptado aos problemas de *designs* físicos, foi projetado para funcionar em conjunto com uma plataforma paramétrica, tal como o *Grasshopper*. O *Discover* consiste em uma biblioteca modular para

otimização mono e multiobjetivo, escrita em *Python*, e uma interface escrita em *JavaScript* que permite explorar visualmente o processo de otimização.

Embora o *Discover* seja uma boa ferramenta para resolver problemas complexos de otimização de projeto, o algoritmo é conduzido por apenas quatro operadores básicos, baseados na seleção natural: (1) geração – o algoritmo gera uma amostra aleatoriamente dos *designs* do espaço de projeto formando a geração inicial; (2) seleção ou ranking – o algoritmo seleciona os indivíduos melhor posicionados para serem usados na próxima geração, criando assim, um "*pool* de acasalamento" contendo os melhores *designs,* de acordo com seu objetivo; (3) cruzamento - os algoritmos selecionados como projetos promissores são recombinados para criar uma nova população de projetos; (4) mutação – assegura que a busca explore soluções não contidas nas gerações passadas por meio de variação pseudorrandômica.

Este caminho é bastante semelhante ao da evolução natural. Em ambos, as informações genéticas dos genitores são aleatoriamente recombinadas para criar um novo indivíduo. A ideia básica é que, uma vez que os genitores sobreviveram tempo suficiente para se reproduzirem, ambos devem ter algum material genético que possa ser útil para a sobrevivência da espécie em geral. Ao se recombinarem, provavelmente o novo individuo herdará características vencedoras de ambos os pais (fator de convergência), aumentando suas chances de sobreviver e de se reproduzir em uma próxima geração.

Entretanto, apenas com esses métodos, podemos ficar sem a melhor solução. Isso acontece quando os projetos da primeira geração, realizada aleatoriamente, apresentam nenhuma ou poucas características fundamentais em sua genética, impedindo a transferência de informações importantes para a nova geração. Então, como ocorre na natureza, precisamos de um mecanismo que possa inserir novas informações de forma aleatória ao fundo genético. Isso é realizado pelo operador de mutação, responsável pela alteração das entradas de um número aleatório de filhos, normalmente uma pequena porcentagem, antes de entrar na próxima geração (fator estímulo à diversidade).

No editor de texto *Sublime 2* abrimos o *template Run*, disponibilizado como parte do pacote *Discover*. Neste *template*, configuramos todos os parâmetros de entrada (linha 04). A seguir, tem-se o exemplo do primeiro *input* preenchido (linha 05):

```
[01] from src import job
[03] jobDescription = {
[04]     jobName: untitled,
[04]     "inputsDef": [
[05]         {"name": "input00","type": continuous",
              "range": [10,20]},
[06] ...
[30]                 ],
```

Neste mesmo arquivo, também definimos os parâmetros de saída (linha 31). Nestas linhas, são definidas as funções objetivas do experimento. Na primeira etapa do experimento, os problemas que nos interessam são aqueles com apenas uma função objetiva, ou seja, aqueles descritos como mono-objetivo, neste caso, a minimização do peso;

```
[31] "outputsDef": [
[32]     {"name": "weight",  "type": objective,
          "goal": "min"},
```

Para a segunda etapa, introduzimos uma segunda função objetiva: maximizar o potencial energético da cobertura (linha 33).

```
[33]     {"name": "powerOutput",
          "goal": "max"},
```

Esse novo objetivo se soma ao anterior, transformando-o em um problema multiobjetivo. Para atender esta nova função, componentes do *plug-in Ladybug*, de analise ambiental, foram aplicados para importar as configurações de módulo fotovoltaico específico da biblioteca *California Energy Commission (CEC) Modules* e para calcular a quantidade de energia elétrica que pode ser produzida pela superfície. O módulo escolhido para esta fase foi a célula *Centrosolar America VS-160C1*, versão NRELv1. Para consideração do experimento, a escolha do modelo levou em conta apenas algumas das características dos modelos, como por exemplo, a flexibilidade e o baixo peso.

Tabela 1: Característica da célula fotovoltaica escolhida

ID[1]	A_C (m²)[2]	PTC[3]	Tecnologia[4]
2385	1.086	148.6	*Thin Film*

[1] Identificação na planilha da *CEC_Modules;*
[2] Área de abertura;
[3] *Standard Test Conditions*
[4] Tipo de tecnologia da célula

Definidos os parâmetros de entrada e saída, preenchemos as opções necessárias para o trabalho geral, que são: (1) número de gerações: 100 (linha 40); (2) número de população: 59 (linha 41); (3) taxa de mutação: 0.05 (linha 42) e (4) número de elite salvas: 10 (linha 43).

```
[38] algo: GA,
[39] "algoOptions": {
[40]     "numGenerations": 150,
[41]     "numPopulation": 59,
[42]     "mutationRate": 0.05,
[43]     "saveElites": 10,
[44]     "DOE": random,
```

Preenchido o *template*, executa-se o trabalho (linha 53).

```
[52]#job.createInputFile(jobDescription)
[53]#job.run(jobDescription)]
```

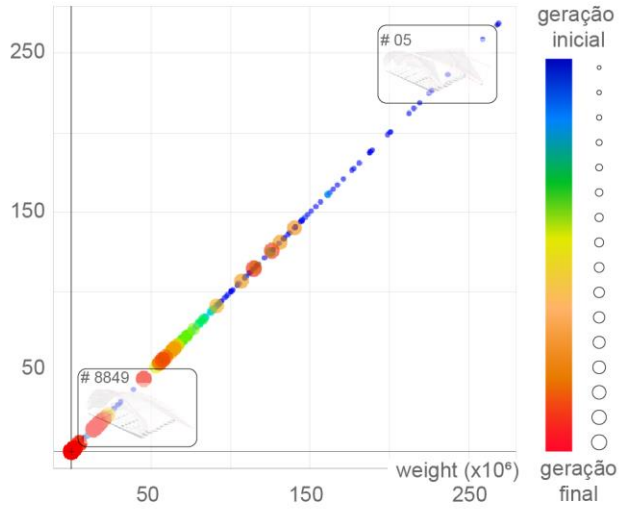
Quando a otimização estiver encerrada, podemos visualizar os resultados plotando-os em relação aos objetivos, em um gráfico de dispersão.

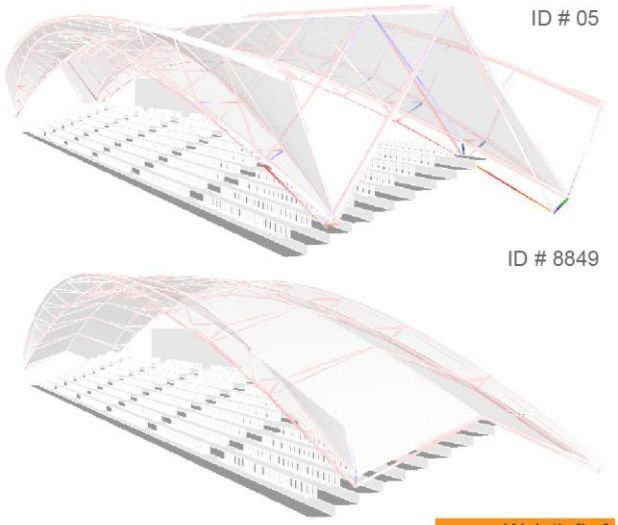
RESULTADOS

Na primeira etapa foi gerado um conjunto de 8.850 projetos. A tarefa neste estágio foi filtrar os conjuntos de dados pelas pontuações e selecionar o projeto com maior

desempenho. Na Figura 3a apresenta-se o gráfico de dispersão do espaço de projeto testado pelo *Discover*. Cada círculo representa um projeto. Nota-se que por ser uma otimização mono-objetiva, o gráfico apresenta todos os projetos alinhados. Isso torna sua interpretação bastante intuitiva, pois quanto menor o peso do projeto mais próximo do canto inferior esquerdo ele está. Neste caso, o projeto #8849 é considerado como ótimo. Isso porque não tem nenhum outro que seja mais leve do que ele. Para comparação, consideremos o projeto #5. Criado ainda na primeira geração, quando os projetos são completamente aleatórios, os indivíduos tem poucos dos componentes certos para alcançar o objetivo. Ainda por isso, observa-se que este projeto foi reprovado em mais

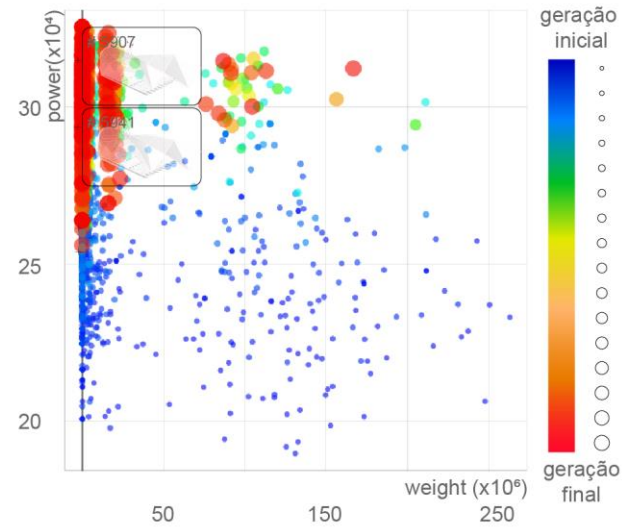
(a)

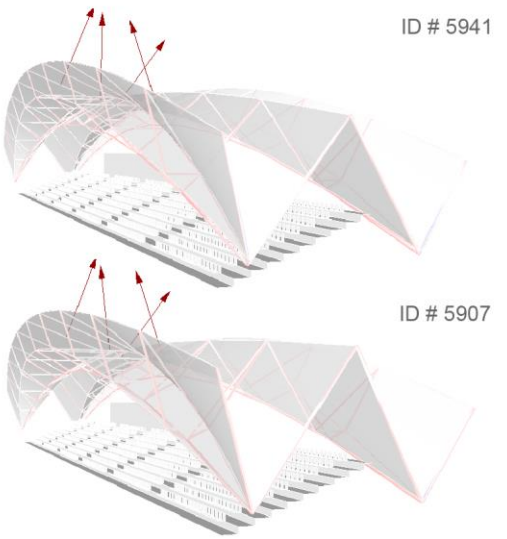




inputs			generation		parent 01		parent 02									output		Weigth [kg]		
ID # 05			0		none		none											268730088.40		
(00)	(01)	(02)	(03)	(04)	(05)	(06)	(07)	(08)	(09)	(10)	(11)	(12)	(13)	(14)	(15)	(16)	(17)	(18)	(19)	(00)
14	6.72	7.00	1.22	2.12	0.71	0.26	4.96	2.76	6.09	5.40	19	16	0.37	0.91	4.32	9.83	9.37	22	28	14
inputs			generation		parent 01		parent 02									output		Weigth [kg]		
ID # 8849			149		8754		8788											9823.62		
(00)	(01)	(02)	(03)	(04)	(05)	(06)	(07)	(08)	(09)	(10)	(11)	(12)	(13)	(14)	(15)	(16)	(17)	(18)	(19)	(00)
10	9.11	9.71	1.03	1.07	0.05	0.17	2.11	2.00	2.06	2.06	10	10	0.03	0.03	2.23	4.05	13	2	2	14

(b)





inputs			generation		parent 01		parent 02									output		Power [kwh]		Weigth [kg]	
ID # 5941			100		5891		5876											300293.98		20900.19	
(00)	(01)	(02)	(03)	(04)	(05)	(06)	(07)	(08)	(09)	(10)	(11)	(12)	(13)	(14)	(15)	(16)	(17)	(18)	(19)	(20)	
10	8.89	8.43	1.59	2.57	0.19	0.36	3.46	4.96	6.77	5.90	19	17	1.19	0.31	4.40	10.0	17	0	0	17	
inputs			generation		parent 01		parent 02									output		Power [kwh]		Weigth [kg]	
ID # 5907			100		5848		none											321819.29		27578.86	
(00)	(01)	(02)	(03)	(04)	(05)	(06)	(07)	(08)	(09)	(10)	(11)	(12)	(13)	(14)	(15)	(16)	(17)	(18)	(19)	(20)	
10	8.32	8.75	2.02	2.93	0.07	0.00	4.44	4.86	6.87	6.95	18	19	1.44	0.06	4.60	9.95	20	0	0	20	

Figura 3: Gráficos de dispersão dos experimentos do processo generativo e projetos selecionados. Fonte: autores.

de um critério na análise estrutural.

Na Figura 3b mostra-se o gráfico de dispersão da segunda etapa do experimento com 8.850 projetos. O objetivo da otimização é encontrar projetos que atendam à restrição estrutural com um peso mínimo e o maior potencial energético possível. Como esses dois objetivos estão em competição um com o outro, não existe uma solução única e sim várias soluções consideradas satisfatórias.

Isolando os projetos de alto desempenho no espaço de projeto, notamos que os desenhos formam uma linha movendo-se do canto inferior esquerdo para cima. Isso porque a estratégia adotada como restrição os deslocou para esta posição. O natural, neste caso, em que a negociação entre ter mais superfície de cobertura contraria o critério do menor peso da estrutura, seria uma linha partindo do mesmo lugar em direção ao canto superior direito. Os projetos ao longo desta linha podem ser considerados todos ótimos. Isso porque melhorar qualquer um desses projetos em um objetivo é torna-lo, necessariamente, pior em outro. Uma vez que esse conjunto de projetos é selecionado, a tarefa passa a ser discutir sobre eles. Aqui, ressalta-se mais uma vez o projeto como processo, entidade aberta à participação dos diversos profissionais de projetos e demais envolvidos, direta e indiretamente, com o espaço a ser construído.

Assim, uma análise profunda de vários projetos de alto desempenho é necessária. Imaginemos uma relação direta do potencial energético como iniciativa para pagar o custo total do aço, em peso, utilizado na construção da cobertura. Destes projetos ótimos, foram selecionados apenas dois (Tabela 2). Temos que o projeto # 5941 é, neste caso, a melhor opção. Mas essa lógica muda quando se tem um planejamento a longo prazo, como por exemplo, a redução de gastos com o consumo energético ao longo de 20 anos. Nesta hipótese temos o projeto # 5907 como melhor opção.

Tabela 2: Comparação de projetos da segunda etapa

ID	kg (R$)[1]	kWh (R$)[2]	Mês[3]	20 anos[4]
5941	418.003,60	176.272,52	28.5	3.107.446,80
5907	551.557,20	188.907,92	35.1	3.226.601,20

[1] Peso total vezes o valor de R$ 20,00
[2] Potencial de energia ano vezes a média do valor kWh R$ 0,587
[3] Peso total dividido por energia/12
[4] Rendimento bruto em 10 anos, excluindo-se a estrutura

DISCUSSÃO

Neste artigo destaca-se o projeto como prática em que arquiteto e projeto dialogam em um fluxo de informações que precedem o projeto em si, orientado por um objetivo comum. Para isso, busca-se avaliar o processo de projeto, discutindo criticamente, a perda de controle como estratégia. Nos experimentos apresentados, essa perda de controle se deu pela modelagem matemática do problema e a busca automática de soluções por meio de heurísticas. À medida que se construiu essa possibilidade, percebeu-se como é frágil a certeza de uma resposta (solução) assertiva para um projeto. Isso acontece porque uma vez que uma decisão tenha sido tomada, acaba-se limitando grande parte do espaço de exploração das possibilidades nas fases posteriores, tornando altamente improvável que o processo de *design* alcance um *design* verdadeiramente "ótimo". Como esses procedimentos estão sendo utilizados há pouco tempo, este trabalho contribui para aqueles que pretendem seguir por este caminho de projeto, apresentando-lhes uma descrição detalhada sobre os procedimentos adotados na: (1) descrição de um modelo de espaço de projeto arquitetônico em aço; (2) definição de um conjunto de diretrizes para avaliar as qualidades de um espaço de projeto baseado na análise estrutural e na capacidade deste de gerar energia fotovoltaica e (3) abordagem de método para avaliar os processos generativos, com mais de um objetivo, por meio de amostragem e visualização.

REFERÊNCIAS

Ashby, W. R. (1970). *Uma introdução à cibernética.* São Paulo: Editora Perspectiva.

Buchanan, R. (1992). Wicked Problems in Design Thinking. *MIT Press Journal* , pp. p. 5-21.

Ferrante, I. P. (2013). *Entre a arquitetura experimental e a arquitetura para a experiência: máquinas, corpos e prótese de Lars Spuybroek/NOX.* São Paulo: Faculdade de Arquitetura e Urbanismo da Universidade de São Paulo .

Glanville, R. (2002). Second order cybernetics. In E. Publishers, *Encyclopedia of Life Support Systems, Systems Science and Cybernetics - Vol. III.* Oxford.

Lawson, B. (1983). *How designers think*. London: Butterworth-Heinemann.

Nagy, D. (2017, jan 23). *Generative Design*. Retrieved from Medium: https://medium.com/generative-design

Oxman, R., & Oxman, R. M. (1992). Refinement and adaptation in design cognition. *Design Studies*, p. 117-134.

Shiffman, D. (2012). *The Nature of Code.* California: Magic Book Project.

Vassão, C. A. (2010). *Metadesign: ferramentas, estratégias e ética para a complexidade.* São Paulo: Editora Blucher.

Villaggi, L., Stoddart, J., Nagy, D., & Beijamin, D. (2018). *Survey-Based Simulation of User Satisfaction for Generative Design in Architecture.* Singapure: Humanizing Digital Reality.

Yang, X.-S. (2010). *Nature-Inspired Metheuristic Algorithms* . Cambridge: Luniver Press.

Yeniay, Ö. (2005). Penalty function methods for constrained optimization with genetic algorithms. *Mathematical and Computational Applications, Vol. 10, Nº 1*, p. 45-56.