UNIVERSIDADE FEDERAL DE MINAS GERAIS

PROGRAMA DE PÓS-GRADUAÇÃO EM CONSTRUÇÃO CIVIL

BIM EVOLUCIONÁRIO: APLICABILIDADE DE ALGORITMOS DE INTELIGÊNCIA ARTIFICIAL NO DESENVOLVIMENTO DE PROJETOS DE EDIFICAÇÕES

Autora: Denise Aurora Neves Flores

Orientador: Prof. Dr. Frederico Gadelha Guimarães

Belo Horizonte

Junho/2021

Denise Aurora Neves Flores

BIM EVOLUCIONÁRIO: APLICABILIDADE DE ALGORITMOS DE INTELIGÊNCIA ARTIFICIAL NO DESENVOLVIMENTO DE PROJETOS DE EDIFICAÇÕES

Versão final

Dissertação apresentada à Escola de Engenharia da Universidade Federal de Minas Gerais como parte dos requisitos para obtenção do título de Mestre em Construção Civil. Área de concentração: Tecnologia na Construção Civil. Linha de pesquisa: Gestão na Construção Civil.

Orientador: Prof. Dr. Frederico Gadelha Guimarães

Belo Horizonte
Escola de Engenharia da UFMG
2021

Flores, Denise Aurora Neves.
F634b Bim evolucionário [recurso eletrônico]: aplicabilidade de algoritmos de inteligência artificial no desenvolvimento de projetos de edificações / Denise Aurora Neves Flores. - 2021.
1 recurso online (xvi,179 f. : il., color.) : pdf.

Orientador: Frederico Gadelha Guimarães.

Dissertação (mestrado) - Universidade Federal de Minas Gerais, Escola de Engenharia.

Anexos: f.170-179.
Apêndices: f. 141-169.
Bibliografia: f: 132-140.

1. Construção civil - Teses. 2. Algoritmos - Teses. 3. Automação – Teses. 4. Inteligência Artificial – Teses. 5. Modelagem de informação da construção – Teses. I. Guimarães, Frederico Gadelha. II. Universidade Federal de Minas Gerais. Escola de Engenharia. III. Título.

CDU: 691 (043)

Ficha catalográfica elaborada pela Bibliotecária Letícia Alves Vieira - CRB-6/2337
Biblioteca Prof. Mário Werneck - Escola de Engenharia da UFMG

UFMG
UNIVERSIDADE FEDERAL
DE MINAS GERAIS

MESTRADO
EM CONSTRUÇÃO CIVIL

ATA DA DEFESA DE DISSERTAÇÃO DE MESTRADO EM CONSTRUÇÃO CIVIL DA ALUNA **DENISE AURORA NEVES FLORES**

Às quatorze horas do dia vinte e quatro de junho de dois mil e vinte e um, reuniu-se, por meio de sistema de interação de áudio e vídeo em tempo real (plataforma Google Meet) a Comissão Examinadora de Dissertação de Mestrado, aprovada *ad referendum* pelo Coordenador do Curso de Mestrado em Construção Civil - EE.UFMG, **Prof. Dr. Eduardo Chahud**, para julgar, em exame final, o trabalho intitulado **"BIM EVOLUCIONÁRIO: APLICABILIDADE DE ALGORITMOS DE INTELIGÊNCIA ARTIFICIAL NO DESENVOLVIMENTO DE PROJETOS DE EDIFICAÇÕES",** requisito final para a obtenção do Grau de MESTRE EM CONSTRUÇÃO CIVIL na área de: TECNOLOGIA NA CONSTRUÇÃO CIVIL. Abrindo a sessão, o Presidente da Comissão, **Prof. Dr. Frederico Gadelha Guimarães** após dar a conhecer aos presentes o teor das Normas Regulamentares do Trabalho Final, passou a palavra à candidata para apresentação do seu trabalho. Seguiu-se a arguição pelos examinadores, com a respectiva defesa da candidata. Logo após, sem a participação da candidata na transmissão, houve o julgamento e expedição do resultado pela Comissão. Foram atribuídas as seguintes indicações:

PROF. FREDERICO GADELHA GUIMARÃES	(X) APROVADA ( ) REPROVADA
PROF PAULO ROBERTO PEREIRA ANDERY	(X) APROVADA ( ) REPROVADA
PROFA. ANDREA QUADRADO MUSSI	(X) APROVADA ( ) REPROVADA

Pelas indicações dos membros acima, a candidata foi considerada APROVADA. Caso tenham sido sugeridas correções no trabalho, conforme pareceres anexos, a candidata terá o prazo máximo de 60 (sessenta) dias para efetuá-las e entregar a versão final da dissertação à Secretaria do Curso de Mestrado em Construção Civil. O resultado foi comunicado a candidata pela Presidente da Comissão. Nada mais havendo a tratar, a Presidente encerrou a transmissão e lavrou a presente ATA, que será assinada eletronicamente por todos os membros participantes da Comissão Examinadora. Belo Horizonte, vinte e quatro de junho de dois mil e vinte e um.

Assinaturas:

Paulo Roberto Pereira Andery:71481214691
Assinado de forma digital por Paulo Roberto Pereira Andery:71481214691
Dados: 2021.07.02 16:11:17 -03'00'

Obs.: Este documento não terá validade sem a assinatura e carimbo da Coordenação do Curso de Mestrado em Construção Civil.

Eduardo Chahud
Assinado de forma digital por Eduardo Chahud
Dados: 2021.07.08 09:04:38 -03'00'

Prof. Dr. Eduardo Chahud
Coordenador do Curso de Mestrado em Construção Civil
ESCOLA DE ENGENHARIA/UFMG

DEDICATÓRIA

Dedico este trabalho a 1,2 bilhão de pessoas no mundo que hoje não têm acesso a moradia e infraestruturas adequadas.

Acredito que somente por meio de tecnologias digitais a Construção Civil poderá ganhar a produtividade e a escala necessárias.

Esta pesquisa é uma pequena contribuição ao conhecimento, em um oceano de necessidades.

AGRADECIMENTOS

Primeiramente, a minha gratidão a Deus pela graça da vida, todo amor e oportunidades que me concedeu.

Especialmente ao meu esposo, André Flores, por me inspirar com sua resiliência e índole. Por seu apoio irrestrito e por me transmitir toda paz para realizarmos esta conquista. Sua generosidade única foi, e sempre será, fundamental.

À minha mãe, Marlene Teodora, por ser minha referência de fé, coragem, trabalho incessante e por seu eterno e essencial cuidado.

Ao meu pai, Jadir Neves, por seu exemplo de amor ao conhecimento, por me inspirar à docência e pelo especial apoio nesta pesquisa, com suas reflexões.

Ao meu irmão e amigo, Daniel Teodoro, por nos iluminar com sua inteligência científica e espiritual, por seu modelo de diplomacia, disciplina e empatia.

À minha cunhada Jacqueline Soncin, por fazer parte de nossa família com seu companheirismo, inteligência e integridade.

Ao meu estimado e distinto orientador, Prof. Dr. Frederico Guimarães, que me concedeu a oportunidade de cursar sua disciplina, por acreditar no meu potencial e na pesquisa, por suas orientações precisas e postura construtiva nesse percurso.

Ao Prof. Dr. Paulo Andery, por sua visão integrada de academia, mercado e ética.

Ao Dr. Roberto Ribeiro, colega do laboratório MINDS, e ao arquiteto Marcelo Azarias, pelas colaborações fundamentais à pesquisa.

À amiga Ma. Maria Luísa Antunes pela parceria, lealdade e ideias nos caminhos da pós-graduação e docência.

Aos amigos e colegas que me apoiaram neste projeto: Alexandre Neves, Fernando Malard e Lúcio Campos.

A todos os docentes que tive durante a vida, por fazerem parte de minha formação técnica e humana.

A todos, minha sincera gratidão.

Um pouco de ciência nos afasta de Deus. Muito, nos aproxima.

Louis Pasteur

RESUMO

No âmbito da indústria da Arquitetura, Engenharia e Construção (AEC), tecnologias paramétricas como o *Building Information Modeling* (BIM) e o *Design* Generativo têm suportado novos sistemas de projetação. O aumento no uso do BIM impactou os estágios de projeto detalhado e construção. A parametrização permitiu diversos avanços na geração, exploração e automação desses projetos. Entretanto, embora os processos generativos e de avaliações automatizadas de requisitos sejam altamente úteis, eles se restringem a funções de execução e não propõem soluções evolutivas. A integração de tecnologias BIM, *design* generativo, verificação automatizada de projetos e algoritmos evolucionários desta pesquisa demonstra contribuir para o incremento da geração otimizada de soluções para projetos de edificações – em especial na fase do *Early Concept Design –,* avançando na aplicabilidade da inteligência artificial no campo da construção civil. Foi adotado o método da pesquisa construtiva, ou *Design Science Research.* O delineamento da pesquisa compreendeu a descrição da problemática e a formulação de um constructo denominado BIM Evolucionário (E-BIM). O desenvolvimento e implementação do constructo ocorreu em três fases: Fase A - desenvolvimento, implementação e avaliação de um algoritmo genético inserido no fluxo de projetos BIM, como instrumento evolucionário; Fase B - geração, verificação e otimização de um objeto arquitetônico, integrando a técnica da Gramática da Forma e as ferramentas computacionais *Rhino-Grasshopper*, *Archicad*, *Solibri Model Checker* e *script Python*; Fase C - verificação da asserção do termo BIM Evolucionário (E-BIM) proposto ao contexto desta aplicação. Os resultados quantitativos apresentados demonstraram: a viabilidade de geração automatizada de soluções em escala; a modelagem automatizada generativa de modelos BIM IFC; a otimização e incremento nos indicadores de desempenho do projeto alcançados com o uso de algoritmos evolucionários. Em conjunto, os resultados apontam que a implementação de um algoritmo evolucionário integrado ao processo de projetos proposto, permite que projetistas explorem, em grande escala, soluções múltiplas e melhorem o desempenho destas soluções, contribuindo para a melhoria da produtividade dos projetos da AEC.

Palavras-chave: Modelagem da informação da construção. *Design* generativo. Automação de projetos. Algoritmo evolucionário. Inteligência artificial.

ABSTRACT

In the scope of the Architecture, Engineering and Construction (AEC) industry, parametric technologies such as Building Information Modeling (BIM) and Generative Design have supported new design systems. The increase in the use of BIM impacted the stages of detailed design and construction. The parameterization allowed for several advances in the generation, exploration and automation of these projects. However, although the generative and automated requirements assessment processes are highly useful, they are restricted to execution functions and do not offer evolutionary solutions. The integration of BIM technologies, generative design, automated design verification and evolutionary algorithms in this research demonstrates that it contributes to the increase of the optimized generation of solutions for building projects - especially in the Early Concept Design phase -, advancing the applicability of Artificial Intelligence in the field of civil construction. The approach Design Science Research was adopted as research methodology. This research design comprised the description of the problem and the formulation of a construct called Evolutionary BIM (E-BIM). The development and implementation of the construct occurred in three phases: Phase A - development, implementation and evaluation of a genetic algorithm inserted in the flow of BIM designs, as an evolutionary instrument; Phase B - generation, verification and optimization of an architectural object, integrating the technique of Shape Grammar and the computational tools *Rhino-Grasshopper*, *Archicad, Solibri Model Checker* and *Python* script; Phase C - verification of the assertion of the term Evolutionary BIM (E-BIM) proposed in the context of this application. The quantitative results demonstrated: the feasibility of automated generation of solutions in scale; the automated generative modeling of BIM IFC models; the optimization and increase in design performance indicators achieved with the use of evolutionary algorithms. Together, the results point that the implementation of an evolutionary algorithm integrated with the proposed design process, allows designers to explore, on a large scale, multiple solutions and improve the performance of these solutions, contributing to improving the productivity of AEC design.

Keywords: Building Information Modeling. Generative Design. Design automation. Evolutionary algorithm. Artificial intelligence.

LISTA DE ILUSTRAÇÕES

LISTA DE QUADROS

LISTA DE TABELAS

LISTA DE ABREVIATURAS E ACRÔNIMOS

ACO	*Ant Colony Optimization*
AEC	Arquitetura, Engenharia e Construção
AG	Algoritmo(s) Genético(s)
AIA	*American Institute of Architects*
API	*Application Programming Interface*
BIM	*Building Information Modeling*
CAD	*Computer Aided Design*
CE	Computação Evolucionária
CNC	*Computer Numerical Control*
DE	*Design* Evolucionário
DG	*Design* Generativo
DNA	*Deoxyribonucleic Acid* ou Ácido Desoxirribonucleico
DSR	*Design Science Research*
E-BIM	BIM Evolucionário
FDE	Fundação para o Desenvolvimento da Educação
GPS	*Global Positioning System*
GSA	*United States General Services Administration*
IA	Inteligência Artificial
IFC	*Industrial Foundation Classes*
ITO	*Information Takeoff*
MOP	*Multiobjective problems*
PG	Programação Genética
PMCMV	Programa Minha Casa, Minha Vida
PMI	*Project Management Institute*
PSO	*Particle Swarm Optimization*
RNA	Rede Neural Artificial
SMC	*Solibri Model Checker*
VPL	*Visual Programming Language*

SUMÁRIO

1 INTRODUÇÃO

1.1 Contexto

O bem-estar de uma sociedade se relaciona, em alguma medida, segundo Bock e Linner (2015), com sua eficiência, maior produtividade e crescimento econômico. Os autores alemães, em uma publicação da Universidade de Cambridge, consideram que tanto processos socioeconômicos quanto sociotécnicos acessíveis e eficientes são necessários ao desenvolvimento. Argumentam sobre as análises de que a produtividade na indústria da Arquitetura, Engenharia e Construção (AEC) vem diminuindo há décadas, em todo o mundo. Além disso, defeitos construtivos, condições de trabalho inadequadas e baixa atratividade da indústria AEC para as gerações mais jovens, bem como o enorme consumo de matérias-primas e energia pelo processo e produtos de construção, apresentam desafios para os quais a construção e arquitetura convencionais atualmente não têm soluções. Assim, os pesquisadores pontuam que a riqueza futura apenas pode ser gerada por saltos de inovação e tipos radicalmente novos de projetos e engenharia de valor.

Diversos e relevantes relatórios de instituições nacionais e internacionais divulgam, a todo tempo, a crescente lacuna em infraestrutura e habitação mundiais. O *World Resources Institute*, no relatório “Rumo a uma cidade mais igual: moradia adequada, segura e acessível” apontou que, àquela altura, a demanda global de moradias populares era estimada em 330 milhões de domicílios urbanos e a previsão de crescimento era de mais de 30%, para 440 milhões de domicílios, o que diz respeito à vida de 1,6 bilhão de pessoas, até 2025. A publicação sinaliza ainda que não somente a carência da habitação precisa ser suprida, mas também carecem de ampliação toda infraestrutura dos serviços como saneamento, mobilidade e equipamentos urbanos públicos (KING *et al*. 2017).

Em muitos cenários, os déficits de infraestrutura e habitação não são ocasionados estritamente por políticas urbanas. Em determinados contextos, tais déficits ainda se relacionam muito diretamente com a questão da improdutividade do setor. A

Nigéria é um destes casos globais. O país tem um déficit habitacional muito grande e crescente, sendo de, aproximadamente, 8 milhões de unidades habitacionais em 1991, chegando a 12 a 14 milhões em, 2007. O problema do déficit habitacional neste caso, é o resultado da escassez de habilidades profissionais, alto custo dos materiais de construção, desafios logísticos, entre outros. Como em todo o mundo, o problema, naturalmente, também se relaciona com uma série de fatores como migração rural-urbana, alto custo dos insumos, ambiente regulatório e legal, estruturas de financiamento habitacional precárias. Porém, neste contexto, especialistas da indústria AEC nigeriana mostraram que, para resolver o problema da habitação inadequada é mandatório e urgente repensar também a limitada diversidade de processos e os sistemas convencionais de construção para um sistema mais flexível e produtivo (KOLO; RAHIMIAN; GOULDING, 2014).

Tullio (2020) corrobora considerando que, para aumento da qualidade de vida das pessoas, os empreendimentos devem atender aos requisitos do trinômio da engenharia: prazo, custo e qualidade, além do conceito de sustentabilidade.

Não suficientemente inquietante o impacto da produtividade da construção no aspecto da sustentabilidade social, Barbosa *et al.* (2017) apresentam uma extensa e minuciosa pesquisa[1] do *McKinsey Global Institute* e *McKinsey Capital Projects & Infrastructure Practice,* onde alertam sobre os impactos econômicos negativos da improdutividade do setor no cenário mundial.

Globalmente, nas últimas duas décadas, o crescimento da produtividade do trabalho na indústria AEC foi em média de apenas 1% ao ano, em comparação com o crescimento de 2,8% para a economia mundial total e 3,6% no caso da indústria de manufatura. Se a produtividade do setor de construção alcançasse a sua eficiência total (produtividade e economia) isso aumentaria o valor agregado do setor em cerca de US$ 1,6 trilhão, o que acrescentaria cerca de 2% à economia

[1] A pesquisa afirma examinar as causas do baixo crescimento da produção na indústria da construção mundial, explora maneiras práticas de melhorar a situação e discute uma mudança em partes do setor em direção a um sistema de produção em massa, padronização, pré-fabricação e modularização - que teriam potencial para aumentar a produtividade em cinco a dez vezes, dependendo do setor.

global, ou o equivalente à metade da necessidade de infraestrutura atual do mundo (BARBOSA *et al.,* 2017).

Vários indicadores sugerem que a metodologia de construção convencional atingiu seus limites. A curva “S” de Foste demonstra a relação entre a estagnação e os limites técnicos do processo convencional e o início, desenvolvimento e crescimento de novas estratégias e tecnologias de automação de construção. Para superar esses limites, o setor deveria fazer maior uso da automação, como outras indústrias de manufatura e serviços já fizeram com sucesso. De toda forma, atualmente, já se pode observar casos em que a tecnologia de automação de construção, sistemas de robôs de serviços e outras tecnologias de microssistemas estão se fundindo com o ambiente construído (BOCK, 2015).

Embora a automação tenha sido usada ativamente e com sucesso em diferentes setores desde a década de 1970, sua aplicação na indústria AEC ainda não é totalmente explorada. Garcia de Soto (2018) realizou uma investigação para avaliar os efeitos da fabricação digital (um tipo de construção automatizada) na produtividade construtiva, analisando o custo e o tempo necessário para a construção de uma parede de concreto complexa. No experimento o autor verificou que a produtividade é maior quando se utiliza o método robótico e percebe que é possível obter também benefício econômico significativo.

Castro-Lacouture (2007), apresenta a avaliação de desempenho de um robô totalmente autônomo que foi usado na construção de pavimento de concreto e suas implicações na produtividade e segurança. Dois processos de pavimentação equivalentes – um convencional e outro automatizado, foram comparados com o uso de ferramentas de simulação, incorporando os recursos necessários para a realização das tarefas e representando as durações com dados de campo e estimativas prototípicas. Os resultados mostraram que o processo automatizado foi mais produtivo, gerando valores até 20% maiores quando simulado por 500 horas, ou 7,5% maiores após atingir o estado estacionário na curva de produtividade *versus* tempo. O processo automatizado utilizou, segundo o autor,

consideravelmente menos mão de obra do que o convencional e maior segurança de operações automatizadas nos ambientes perigosos de construção.

A Universidade Técnica de Munique, dentro do grupo de alta tecnologia da Baviera, considera que o conhecimento integrado em vários campos do conhecimento é necessário para trabalhar com as mudanças complexas que a construção civil irá sofrer na transição de uma indústria de baixo desempenho - baseada em artesanato, para uma indústria de alto desempenho - baseada em máquinas. O setor se expandirá para novos campos do conhecimento, transferindo e absorvendo tecnologias avançadas de várias disciplinas (arquitetura, engenharias civil, mecatrônica, elétrica, de produção, ciência da computação, tecnologia da informação, entre outras). O sucesso desta construção do futuro dependerá da capacidade de lidar com a inovação e de melhorar a cadeia de valor completa, viabilizando e integrando as tecnologias (BOCK; LINNER, 2015).

Em outro relevante relatório do setor, Blanco (2018) afirma que, em se tratando de tecnologias digitais, a inteligência artificial (IA) pode oferecer maneiras novas, mais confiáveis, precisas e rápidas de realizar tarefas também na indústria AEC. O autor coloca exemplos de aplicações alimentadas por IA que, originadas em outras indústrias, podem ser transferidas para a construção: algoritmos de otimização de rotas de transporte para otimização do planejamento do projeto; previsão de resultados farmacêuticos para questões de construtibilidade; otimização da cadeia de suprimentos de varejo para gerenciamento de materiais e estoque; robótica para construção modular ou pré-fabricada e impressão em 3D; reconhecimento de imagens de saúde para gerenciamento de riscos e segurança. Assim, a IA aplicada à indústria AEC pode ter um alcance variado, desde melhorias incrementais de produtividade a novos paradigmas de gerenciamento e projetos da construção.

Uma pesquisa da *AI Innovators*: *Cracking the Code on Project Performance* (2019), do *Project Management Institute* (PMI) revela que 81% dos entrevistados relatam que sua organização está sendo impactada pelas tecnologias de IA e 37% dizem que adotar essas tecnologias é uma alta prioridade para sua organização.

1.2 Problema de pesquisa

Bock e Linner (2015) afirmam que, para ser capaz de desenvolver plenamente seu potencial, a construção do futuro necessita de uma mudança disruptiva em toda a sua cadeia. O que, para eles, inclui: modelos de negócios, organização, produtos, processos e projetos.

No âmbito dos processos de projetos de construção, o projeto digital integrado resulta em mais controle e flexibilidade nas fases de construção, permitindo que mais simulações, avaliações e ajustes sejam feitos em estágios anteriores, sem aumentar os custos de construção (GARCIA de SOTO, 2018).

As tecnologias paramétricas têm suportado novos sistemas de projetação na arquitetura e engenharia, tais como o *Building Information Modeling* (BIM) e o Projeto ou *Design* Generativo. O BIM, por exemplo, pode ser compreendido como um protótipo virtual do edifício, no qual diversas simulações podem ser realizadas para um melhor planejamento e controle do processo real (EASTMAN *et al.*, 2014).

O projeto paramétrico é um método em que as soluções de arquitetura e engenharia são definidas por meio de especificação de seus parâmetros. Os elementos das formas que caracterizam os projetos são expressos por meio desses parâmetros e da variação deles. Dessa maneira, cada alteração realizada sobre um componente paramétrico do conjunto resultará numa forma diferente. Assim, a partir da modificação de um ou mais parâmetros é possível obter facilmente soluções diversas (BARRETO, 2016).

O *Design* Generativo (DG), conforme Barreto (2016), é um método que produz formas por meio de algoritmos, permitindo mecanizar tarefas, produzir geometrias complexas, e otimizar a criação em *design*. Ferramentas de DG são disponibilizadas através de linguagem de programação visual.

A tecnologia BIM é fundamentalmente paramétrica e o *Design* Generativo é algorítmico e paramétrico. Se a inteligência que embasa a multiplicação ou a

otimização de alternativas de projeto é dada pelos algoritmos, os valores fundamentais que variam alimentando estas regras e alterando os resultados são os parâmetros – meio pelo qual plataformas viabilizam projetos mais ágeis e inteligentes (DAVIS, 2013).

Desde o início dos anos 1970, quando a representação eletrônica do projeto de construção se tornou disponível, a verificação automatizada de regras de construção tornou-se também um foco de interesse e estudo na automatização de projetos. Entende-se que esse processo automatizado facilita a avaliação do projeto, reduzindo o tempo e custos de sua avaliação e aumentando a objetividade e a confiabilidade destas análises. Graças à disseminação do BIM, já é possível gerar modelos virtuais paramétricos, e vários sistemas automatizados de verificação de código de construção estão sendo desenvolvidos e aplicados (SOLIBRI, 2016).

Na atual coordenação e análise dos modelos paramétricos BIM, após o relatório gerado no *software* de checagem, a solicitação de revisão do projeto é requerida ao projetista daquela autoria. A partir de métodos individuais e de seu conhecimento tácito, o autor do projeto reformula as inconsistências encontradas e submete o projeto a uma nova rodada de checagem das regras, e, assim, o ciclo se repete sucessivamente, até que se esgotem os conflitos.

Especialistas humanos confiam em suas experiências anteriores e se restringem a estas experiências na nova rodada de soluções, sem qualquer “aconselhamento” de algoritmos. Assim, embora altamente útil na automação de verificação de parâmetros e na comunicação das inconformidades verificadas, o SMC, como *software* de avaliação, se restringe à função analítica e não propositiva na análise dos projetos de engenharia e arquitetura. Em outras palavras, observa-se que projetos paramétricos permitem ser automatizados, mas não se baseiam em inteligência de dados.

Segundo Domingos (2017), subcampos da IA se dedicam ao desenvolvimento de algoritmos e técnicas que permitam ao computador aprender, isto é, aperfeiçoar

seu desempenho em alguma tarefa. O princípio de aplicabilidade destes algoritmos é que, no lugar de programar todas as situações possíveis por meio de instruções de computador, a máquina é treinada com os dados e “aprende” a maneira de executar uma determinada tarefa.

Considerando-se que o ambiente computacional dos projetos paramétricos já seja uma realidade disponível na indústria da construção é possível imaginar que algoritmos de IA possam não apenas nos ajudar a ser mais rápidos e eficientes, mas também a incrementar nossa criatividade e experiência. Um algoritmo pode então processar e analisar dados orientados a gerar projetos novos e com melhor desempenho (NAGY, 2017).

A Computação Evolucionária (CE) é um ramo de pesquisa emergente da inteligência artificial que compreende um conjunto de técnicas de busca e otimização inspiradas na evolução natural das espécies. Desta forma, cria-se uma população de indivíduos (soluções para um problema) que vão reproduzir e competir pela sobrevivência. Os melhores sobrevivem e transferem suas características a novas gerações (EIBEN; SMITH, 2008).

Algoritmos Genéticos (AG) são provavelmente os algoritmos de busca da computação evolucionária mais conhecidos e aplicados. O conceito foi proposto por John Henry Holland, em 1975, e o objetivo era explicar os processos adaptativos dos sistemas naturais e projetar sistemas artificiais baseados neles (DAVIS, 1991).

A abordagem denominada *Design* Evolucionário (DE) diz respeito à aplicação dos Algoritmos Evolucionários ao *Design* Generativo. Alguns arquitetos e pesquisadores exploram as possibilidades da expressão de conceitos arquitetônicos como regras geradoras, para que sua evolução e otimização possam ser aceleradas e testadas pelo uso de modelos computacionais (BENTLEY, 1999).

Embora o DE tenha sido explorada desde o início do desenvolvimento do *design* computacional, ainda é um vasto campo para exploração. Além disto, conforme a

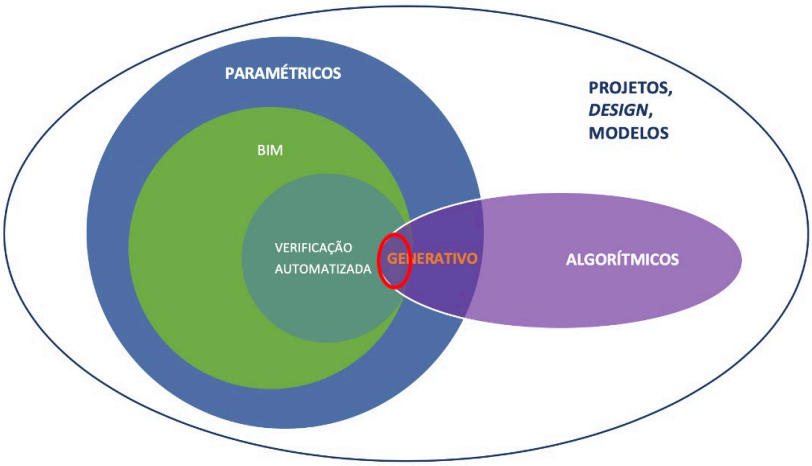
revisão bibliográfica desta pesquisa apresenta, não foram encontrados registros desta aplicabilidade considerando-se também os projetos paramétricos BIM e avaliação automatizada de projetos.

> Os problemas de planejamento espacial com base em uma abordagem evolucionária foram desenvolvidos por 22 anos e ainda não foram totalmente explorados. O desenvolvimento da computação nos últimos anos não impacta significativamente o desenvolvimento de problemas de planejamento de espaço e novos algoritmos para otimizar o tempo de processamento são necessários para permitir que o número de elementos no arranjo seja aumentado. Problemas de planejamento de espaço constituem um amplo campo de pesquisa e têm sido explorados desde o início do desenvolvimento da área de *design* computacional, embora novos métodos e o aprimoramento dos antigos tenham que ser feitos para o desenvolvimento desse campo (CALIXTO; CELANI, 2015, p.669).

Observa-se assim, que mesmo projetos de construções que já sejam praticados em ambiente paramétrico, necessitam ainda avançar para o incremento da inteligência artificial, como tantos outros campos do conhecimento. No futuro, o aumento dessa aplicabilidade poderá elevar as soluções de projetos (criação, avaliação e otimização) ao nível da indústria AEC que urge se reinventar.

Desta forma, promover a integração e sinergia das diversas tecnologias de projetos - projetos paramétricos BIM, design generativo, verificação automatizada de projetos e algoritmos evolucionários é promover o incremento na cadeia da construção do futuro como previu Bock e Linner (2015).

Figura 1 - Diagrama conceitual de projetos e modelos virtuais



Fonte: A autora (2020).

1.3 Questões de pesquisa

Conforme o problema de pesquisa discutido anteriormente, é definida a seguinte questão principal:

“Como associar as tecnologias paramétricas digitais disponíveis a algoritmos de IA, como contributo a uma construção do futuro mais eficiente e produtiva?”.
A questão principal se desdobra nas seguintes questões secundárias:

1) Como associar as tecnologias paramétricas digitais disponíveis a algoritmos de IA, como contributo a uma construção do futuro mais eficiente e produtiva?

2) A integração das estratégias de geração, avaliação e evolução de uma solução de projeto paramétrico é viabilizada computacionalmente?

3) Há incremento nos indicadores de desempenho das soluções de projetos produzidas generativamente após a implementação do algoritmo evolucionário?

4) Verifica-se a asserção do termo BIM Evolucionário (E-BIM) no contexto dos algoritmos de inteligência artificial aplicados a projetos de engenharia e arquitetura? Quais indícios sustentariam essa premissa?

1.4 Objetivos

O objetivo geral da presente pesquisa é propor, viabilizar e analisar a aplicabilidade e incremento de algoritmos de inteligência artificial no processo de elaboração dos projetos de edificações, associando as tecnologias de projetos paramétricos BIM, *design* generativo, verificação automatizada de projetos e algoritmos evolucionários. Foram também definidos os seguintes objetivos específicos:

a) desenvolver, implementar e avaliar um algoritmo genético inserido no fluxo de projetos BIM, como instrumento evolucionário;

b) gerar, avaliar e evoluir (otimizar) um objeto arquitetônico integrando as ferramentas computacionais *Rhino-Grasshopper*, *Archicad*, *Solibri Model Checker* e *script Python*;

c) verificar a asserção do termo BIM Evolucionário (E-BIM) proposto ao contexto dos algoritmos de inteligência artificial aplicados a projetos paramétricos de edificações.

1.5 Delimitações

A pesquisa propõe desenvolver um constructo aplicável a contextos de projetos de edificações e, sob o ponto de vista acadêmico, a pesquisa contribui com uma lacuna de conhecimento a respeito dos problemas de planejamento espacial assistidos por algoritmos e inteligência e tecnologias projetuais. Contudo, devem ser consideradas as premissas e delimitações da investigação:

a) embora a problemática dos déficits habitacional e de infraestrutura tenha sido mencionada na contextualização deste trabalho para elucidação de correlações da improdutividade da indústria AEC, o objeto de investigação e a prova de conceito não se direcionam a nenhuma tipologia específica.
b) as simulações do constructo ocorreram em ambiente simulado com problemáticas e condicionantes alusivas a problemáticas reais.

1.6 Estrutura do trabalho

Para cumprir os objetivos, o trabalho foi estruturado nos seguintes capítulos: no capítulo um é realizada a introdução ao tema, com a definição do problema de pesquisa, descrição dos objetivos geral e específicos e da estruturação do texto; no capítulo dois é apresentada uma revisão bibliográfica sobre os diversos conceitos necessários ao aprofundamento de conhecimento teórico e prático do problema e da solução proposta; o capítulo três descreve detalhadamente o método científico utilizado na pesquisa, incluindo sua estratégia e delineamento; o quarto capítulo contempla a descrição das etapas realizadas no desenvolvimento e aplicação do constructo proposto; no capítulo cinco são apresentados os aspectos dos resultados obtidos na aplicação, bem como discutidas as percepções destes resultados; o sexto capítulo é composto pelas conclusões e sugestões de trabalhos futuros.

2 REVISÃO BIBLIOGRÁFICA

A fundamentação teórica para a construção da pesquisa selecionou fontes de informação sobre bases de periódicos, dissertações e teses nacionais e internacionais.

2.1 Projetos paramétricos

Na matemática, as Equações Paramétricas são comumente usadas para expressar as coordenadas dos pontos que compõem um objeto geométrico, como uma curva ou uma superfície. Nesses casos, o conjunto destas equações é chamado de parametrização do objeto (STOVER; WEISSTEIN, 2019).

A definição de parametrização também é compreendida como:

> Processo que se baseia na definição de relações topológicas entre as partes de um modelo e seus parâmetros variáveis. A partir da variação destes parâmetros, o modelo é alterado, mantendo-se as relações topológicas estabelecidas. Normalmente, é um processo matemático, baseando-se em algoritmos e equações (CAMPOS, 2017, p.13).

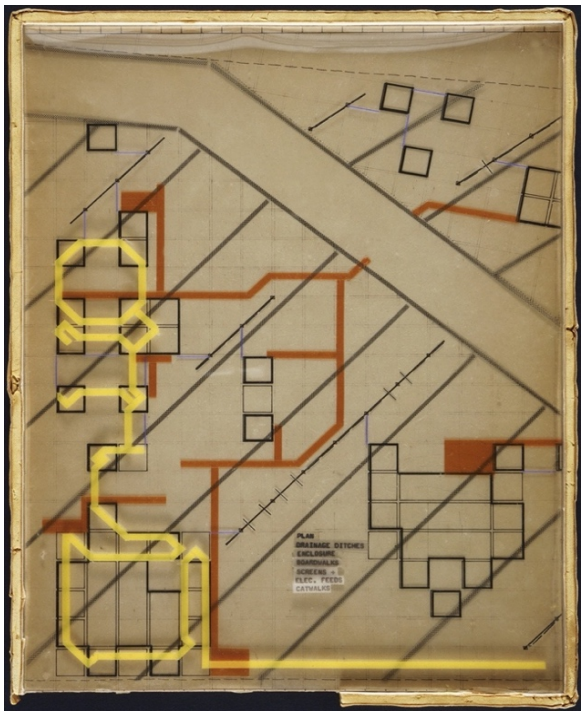
Para Kolarevic (2005), a arquitetura (ou modelagem) paramétrica seria aquela que usa a parametrização como base técnica para gerar formas tridimensionais complexas.

Davis (2013) alerta para uma importante distinção entre Modelagem Paramétrica e Modelagem 3D. Enquanto a Modelagem Paramétrica é baseada na inserção de funções e variáveis, e, assim, manipula a forma a partir da alteração de parâmetros, a Modelagem 3D é construída baseada exclusivamente na manipulação de elementos geométricos tridimensionais.

A ideia da modelagem paramétrica computacional foi a característica essencial do primeiro programa de desenho assistido por computador, ou *computer aided design* (CAD), o Sketchpad, desenvolvido em 1963 por Ivan Sutherland no *Massachusetts*

Institute of Technology. O princípio do programa era o de que, para gerar um objeto, essa geometria possuísse atribuições de um conjunto de parâmetros. Em meados da década de 1980, *softwares* baseados em 3D paramétrico, tais como o CATIA, SolidWorks, Pro/Engineer e outros, alcançaram uma presença comercial. Alguns tornaram-se padrão em setores como eletrônica, infraestrutura, engenharias naval e aeroespacial e fabricação de automóveis. Entretanto, o setor AEC ainda implementaria sistemas de desenho assistido por computador, ou *computer aided design* (CAD), 2D ou as modelagens 3D não paramétricas. O *software* AutoCAD foi uma destas ferramentas mais adotadas. Demorou mais de duas décadas para que o modelo paramétrico 3D fosse adotado no segmento das engenharias e arquitetura (CELANI, 2011; DAVIS, 2013).

Figura 2 - *The Generator Project* - 1976, de Cedric e Frazer



Fonte: Arquivo *online* MOMA (2002).

O AutoLISP – uma linguagem de programação projetada para adaptar a funcionalidade do AutoCAD – foi introduzida pela primeira vez em meados da década de 1980, como uma interface de programação de aplicativos, ou *Application Programming Interface* (API), no AutoCAD Versão 2.1. O aplicativo pode controlar todas as operações do AutoCAD, sendo muito útil para criar mais ferramentas que

aumentam a produtividade do AutoCAD e também permitir a exploração de geometrias mais complexas, que as ferramentas CAD convencionais não são capazes de produzir (AUTODESK, 2012).

Entretanto, Celani (2011) aborda que para profissionais técnicos da engenharia e arquitetura é necessário um grande esforço para utilizar o potencial das linguagens de programação. Assim, o AutoLISP e outras linguagens de *script* não foram amplamente aderidas no setor, pois requerem uma habilidade de programação que a maioria destas partes interessadas não possui, até os tempos atuais.

Eastman *et al.* (2014) abordam ainda que, apesar de setores de manufatura, como a Eletrônica, terem alcançado a parametrização nos anos de 1980, o mesmo não foi possível para o setor AEC, pois, ao contrário do que ocorria na indústria, a tecnologia computacional naquela ocasião não era tão disponível, ou viável, para a complexidade e escala dos componentes de uma edificação se comparados aos produtos manufaturados.

2.1.1 Building Information Modeling (BIM)

Os programas de modelagem paramétrica precisavam de computadores de alta capacidade, por isso os usos foram limitados, durante décadas, à indústria de manufatura, e seu desenvolvimento ficou estagnado até que a capacidade de processamento computacional crescesse a um patamar compatível à escala e complexidade de edificações. Durante este período, diferentes tipos de pesquisas foram realizados para continuar a ampliar as alternativas nas operações de modelagem de sólidos (SACKS *et al.,* 2004).

Em 1975, um conceito muito semelhante ao que compreendemos atualmente como Modelagem da Informação da Construção ou *Building Information Modeling* (BIM) foi tratado, pela primeira vez, por Chuck Eastman, no trabalho *"Building Description System*", publicado no *American Institute of Architects* (AIA) *Journal*. Já o primeiro uso do termo *"Building Modeling"*, no sentido de BIM, foi publicado em um artigo de Robert Aish, em 1986. A publicação descrevia, então, um estudo de caso onde o

sistema de modelagem de edifícios foi aplicado, ilustrando os argumentos e conceitos de BIM conforme a abordagem que conhecemos hoje (QUIRK, 2012; EASTMAN *et al.*, 2014).

Ao contrário do CAD, no lugar de produzir modelagem 3D simples, o BIM modela objetos paramétricos contendo todas as informações necessárias para a construção. Essas informações podem ser extraídas do modelo BIM a qualquer momento durante o processo do projeto, incluindo planos, materiais, dimensões, estimativas de custo, cronogramas e análises de projetos. Além disso, essas informações servem para apoiar todas as partes envolvidas nos processos de projeto e construção, de especialistas, empreiteiros, construtores, clientes e até mesmo a equipe de manutenção pós-construção. Outra vantagem do BIM é a precisão e consistência do projeto. Como trabalha-se com um modelo paramétrico 3D e todas as visualizações são interconectadas, qualquer alteração em uma parte do projeto será atualizada automaticamente em todas as visualizações (EASTMAN *et al.*, 2014).

Assim, no conceito BIM, um modelo tridimensional preciso do edifício é construído digitalmente com o uso de informações paramétricas. A principal característica do que o distingue das tecnologias de projeto que o precedem não é a modelagem tridimensional, mas a informação sistematizada, que pode ser organizada, definida e permutável (SMITH; TARDIF, 2009).

Desenvolvido por Gabor Bojar, na Hungria, o primeiro *softwar*e BIM verdadeiro foi o Radar CH, que mais tarde se tornou o Archicad. Embora tenha sido o primeiro *software* BIM do mundo, devido às limitações de computação da época e a questões de mercado, ele só se tornou popular nos últimos anos. Então, nos anos 2000, foi desenvolvido e amplamente impulsionado o *software* Revit, trazendo muitas inovações que revolucionaram o setor, como famílias paramétricas, controle de fases de construção, cronogramas e ambiente de programação visual (QUIRK, 2012).

Embora totalmente suportado pela tecnologia computacional, o BIM pode ser abstraído em três níveis: (1) como um produto, refere-se a um modelo virtual da edificação; (2) como ferramenta, refere-se às aplicações que interpretam o modelo e a ele agregam informações e representações; (3) como um processo, o BIM é formado pelas atividades desenvolvidas durante todo o ciclo de vida da edificação. Autores como Chuck Eastman, Rafael Sacks e Bilal Succar estudaram a ampla aplicabilidade do BIM, fazendo a associação entre seus usos e benefícios, localizados nas fases do ciclo de vida de um edifício (MANZIONE, 2013).

Quadro 1 - Usos BIM

PROJETO	**Visualização**	Projetos com visualização em 3D Controle de ciclos de revisões Documentação e detalhamento Escaneamento de edifícios com raio laser Fotogrametria Representação realística Realidade virtual Realidade aumentada
	Análise	Verificações de requisitos de normas Estimativas de custo Análises estruturais por elementos finitos Simulação de fogo e fumaça Análises de luminotecnia Levantamentos quantitativos Análises de implantação no terreno Estudos de radiação solar Coordenação espacial e análise de interferências Análise estrutural Análises de sustentabilidade Análises energéticas Análises térmicas Estudos do impacto do vento
CONSTRUÇÃO	**Execução**	Consuntibilidade Construção virtual Segurança do trabalho Especificações da construção Projeto de sistemas construtivos Tecnologias móveis para uso no canteiro Planejamento e controle da produção Licitações e contratações
	Pré-fabricação	Estruturas metálicas Estruturas em concreto pré-moldado
	Aquisição	Coordenação dos suprimentos Preparação de pacotes de compras
OPERAÇÃO	**Gerenciamento**	Rastreamento dos ativos Manutenção dos ativos Monitoramento de ativos por GPS

		Gerenciamento dos espaços Gerenciamento de reformas
	Simulação	Gestão dos sistemas Planejamento para situações de emergência Análises do consumo energético Rastreamento da ocupação
OTIMIZAÇÃO DE PROCESSOS		*Lean* construction Gestão da cadeia de suprimentos Gestão do conhecimento Análises de valor Melhoria do processo de comunicação

Fonte: Manzione (2013).

Visto a amplitude e aplicabilidade do conceito do BIM, tem-se que seus fundamentos consideram questões relativas não somente à parametrização, mas também dizem respeito à interoperabilidade e à colaboração entre os diversos profissionais deste setor. Assim, esse processo envolve distintas fases e especialistas ao longo de todo o ciclo de vida do projeto. Ocorre, entretanto, que são verificadas dificuldades na troca da informação, devido à baixa interoperabilidade atual, emergindo, portanto, um dos fatores limitantes do uso do BIM atualmente (RUSCHEL e ANDRADE, 2009).

Desta forma, mesmo com as dificuldades de interoperabilidade, a *BuildingSMART* International, como discorre Manzione (2013), empenha-se na colaboração irrestrita entre os *stakeholders* BIM, estabelecendo um padrão neutro para troca de dados entre aplicativos, permitindo a representação de toda a edificação em um modelo numérico, através da especificação de estruturas de dados chamadas classes. Esse padrão foi denominado como *Industry Foundation Classes* (IFC) e trata-se de um modelo semântico de dados formado por constructos que representam os diversos objetos da edificação, as suas propriedades, comportamentos e relacionamentos com outros objetos.

Para Ruschel e Andrade (2009), se existe uma boa interoperabilidade entre os aplicativos, é dispensada a necessidade de réplica de dados de entrada (que já tenham sido gerados), viabilizando, de forma automatizada e sem obstáculos, o fluxo de trabalho entre diferentes aplicativos, durante o processo de projeto. Porém, um dos maiores obstáculos para a adoção do IFC é a própria perda de robustez e confiabilidade dos dados na interface entre os aplicativos. A argumentação propõe

também que o uso de aplicativos BIM ainda aparece como um evento fragmentado, limitando a integração da potencialidade dos diversos aplicativos, de vocações tão próprias.

Outras desvantagens das ferramentas BIM percebidas por Caetano *et al.* (2016) são as tarefas repetitivas e bastante morosas, necessárias na modelagem, principalmente na produção de geometrias de crescente complexidade.

Andia e Spiegelhalter (2014) ponderam, então, que apesar das alegações exageradas de que o BIM está revolucionando o setor de AEC, observa-se que este é, ainda, em determinados momentos do processo, um procedimento intensivo em mão de obra, não sendo um método computacional totalmente automatizado, tampouco inteligente[1]. Já Faceli *et al.* (2011) consideram que a capacidade de aprendizado é essencial para um comportamento inteligente.

Contudo, como visto inicialmente neste capítulo, os aplicativos BIM têm várias vantagens em relação às ferramentas CAD tradicionais. Como tal, Barreto (2016) considera estimulante explorar o *Design* Generativo em associação à tecnologia BIM.

2.1.2 Design generativo (DG)

O *Design* generativo (DG) é uma abordagem de *design* que permite a geração de formas por meio de algoritmos. Devido à natureza algorítmica do DG, é possível resolver a mecanização de tarefas repetitivas e demoradas. Essa automação colabora com os projetistas minimizando erros, permitindo economizar tempo e esforço durante o processo. Essa vantagem é mais perceptível em projetos com alto grau de repetição. Usando o DG, em vez de projetar uma edificação, projeta-se o sistema que projeta a edificação (BARRETO, 2016).

Um sistema generativo, segundo Sedrez e Martino (2018), proporciona a geração de múltiplas alternativas a um dado problema com parâmetros em aberto, a partir

[1] I**nteligência**: sf 1.faculdade de conhecer, compreender e aprender. 2.capacidade de resolver situações novas com rapidez e êxito, adaptando-se a elas por meio do conhecimento adquirido.

da entrada de valores para tais parâmetros. Conjuntos de regras podem ser aplicados em diferentes ordens e combinações, ou por modelos geométricos, formando um sistema generativo.

Silva Júnior (2011) aborda a Gramática da Forma como um sistema generativo capaz de criar formas a partir de regras que determinam relações e operações combinatórias de um dado grupo de elementos, definindo uma linguagem formal. Sistemas Generativos são conhecidos por três principais características: (1) capacidade de geração de múltiplas soluções, viabilizando análises e mais estudos do projeto; (2) ampliação criativa e inovadora dado o grande número de soluções geradas; (3) obtenção dos melhores projetos a partir das precisas definições de um dado problema (FISCHER e HERR, 2001).

Devido às dificuldades apresentadas na interoperabilidade BIM, muitas vezes são desenvolvidas soluções de aplicativos personalizados que se valem de APIs, por fornecerem uma forma programática de interação. Por outro lado, o desenvolvimento de APIs requer amplo conhecimento de linguagens de programação avançada, que poucos arquitetos ou engenheiros possuem, conforme Barreto (2016).

Para solucionar esta complexidade e viabilizar a interface, foram desenvolvidos programas com linguagem de programação visual ou *visual programming language* (VPL). Os programas escritos com linguagem visual representam algoritmos gráficos como redes de nós, sendo parâmetros ou componentes. Enquanto parâmetros, contêm dados numéricos, textuais, geométricos ou lógicos (verdadeiro/falso). Sendo componentes, realizam tarefas como operações matemáticas, geração e transformação geométrica (BUENO, 2016; FERREIRA; LEITÃO, 2016).

Dynamo e Rhino-Grasshopper são alguns destes aplicativos de linguagens de programação simples, baseadas em metáforas visuais. Embora amigáveis para o desenvolvimento de pequenos algoritmos, essas linguagens se tornam uma

barreira para soluções muito complexas em atributos e escala, dificultando a manipulação e leitura da codificação (BARRETO, 2016).

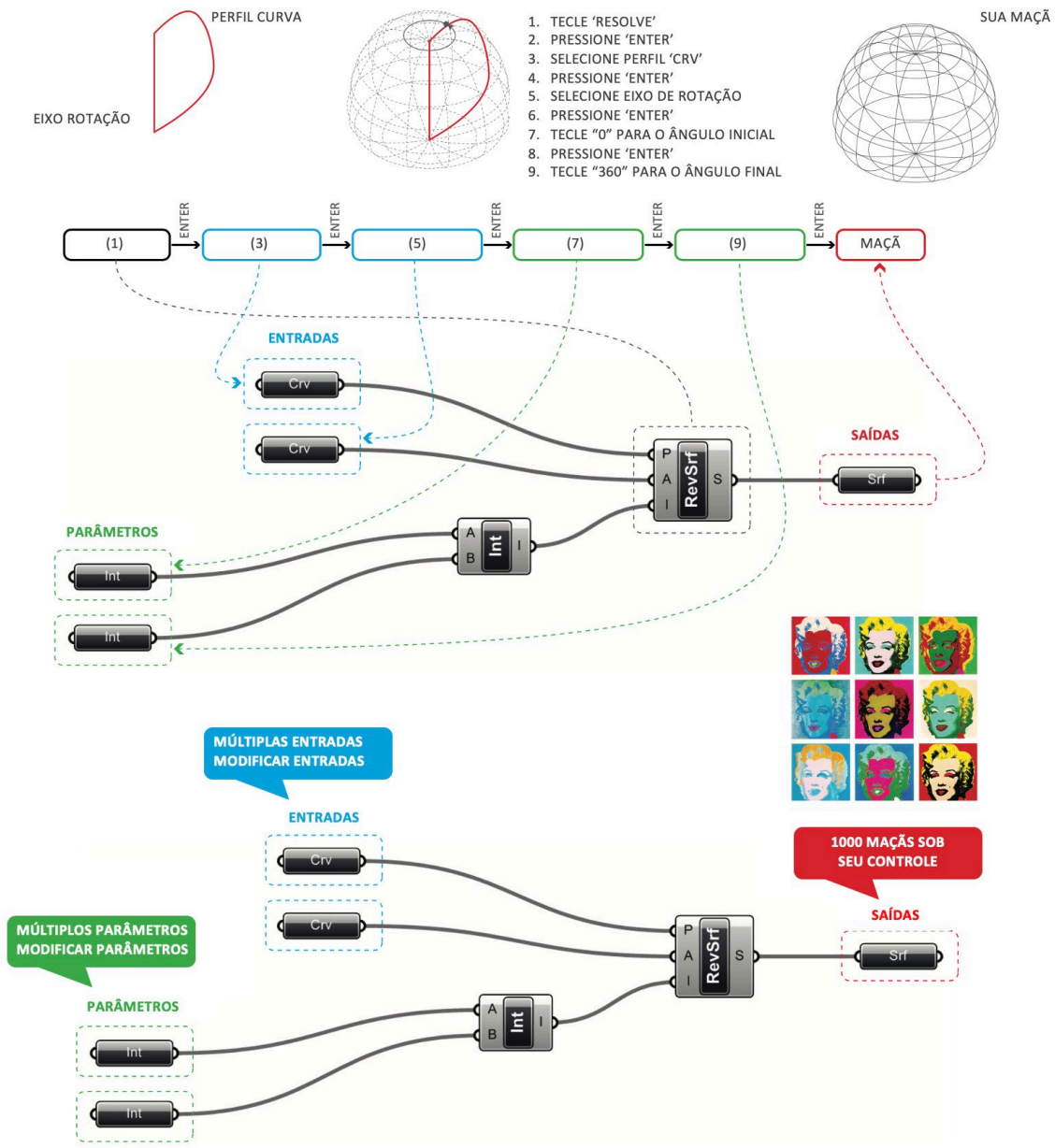
> O Grasshopper rapidamente evoluiu de um sistema de composição explícita do histórico de modelagem para ser um modelador algorítmico com uma linguagem de programação visual, capaz de estabelecer associações paramétricas de projeto arquitetônico, implementar algoritmos generativos e evolutivos e fornecer meios para entrada e saída de dados para outros tipos de *software*, incluindo BIM e dispositivos como *smartphones*, em tempo real (RMA, 2013, s.p.).

Tanto a modelagem paramétrica quanto o *design* generativo são ferramentas para materializar uma solução, e os profissionais escolhem aquela que melhor se adapte às suas necessidades, de acordo com a complexidade e o estágio de desenvolvimento do projeto, tirando partido do melhor de cada ferramenta (SAMMER, 2019).

O Grasshopper é um *plug-in* para o *software* Rhinoceros 3D (*software* CAD), que fornece uma linguagem de programação visual baseada em blocos funcionais que podem ser conectados em uma sequência de ações. Trabalhando de forma integrada, é possível visualizar no Rhinoceros a modelagem 3D programada no Grasshopper. Embora mais intuitivo para programação de *design*, a VPL contém a desvantagem da falta de escalabilidade. Conforme os algoritmos crescem em complexidade, tornam-se de difícil manipulação (FERREIRA; LEITÃO, 2016).

> No Grasshopper, você não trabalha nas geometrias reais. Em vez disso, você trabalha na lógica por trás das geometrias. Uma vez que a lógica de fazer maçã foi configurada, você pode alterar parâmetros como ângulos inicial e final. Além disso, você pode alterar a forma das maçãs fornecendo diferentes curvas de perfil, bem como o eixo de rotação. Ou você pode fazer várias maçãs com apenas uma definição, fornecendo vários valores de ângulo e / ou curvas de perfil / rotação para a definição (SUNG, 2010, p. 4).

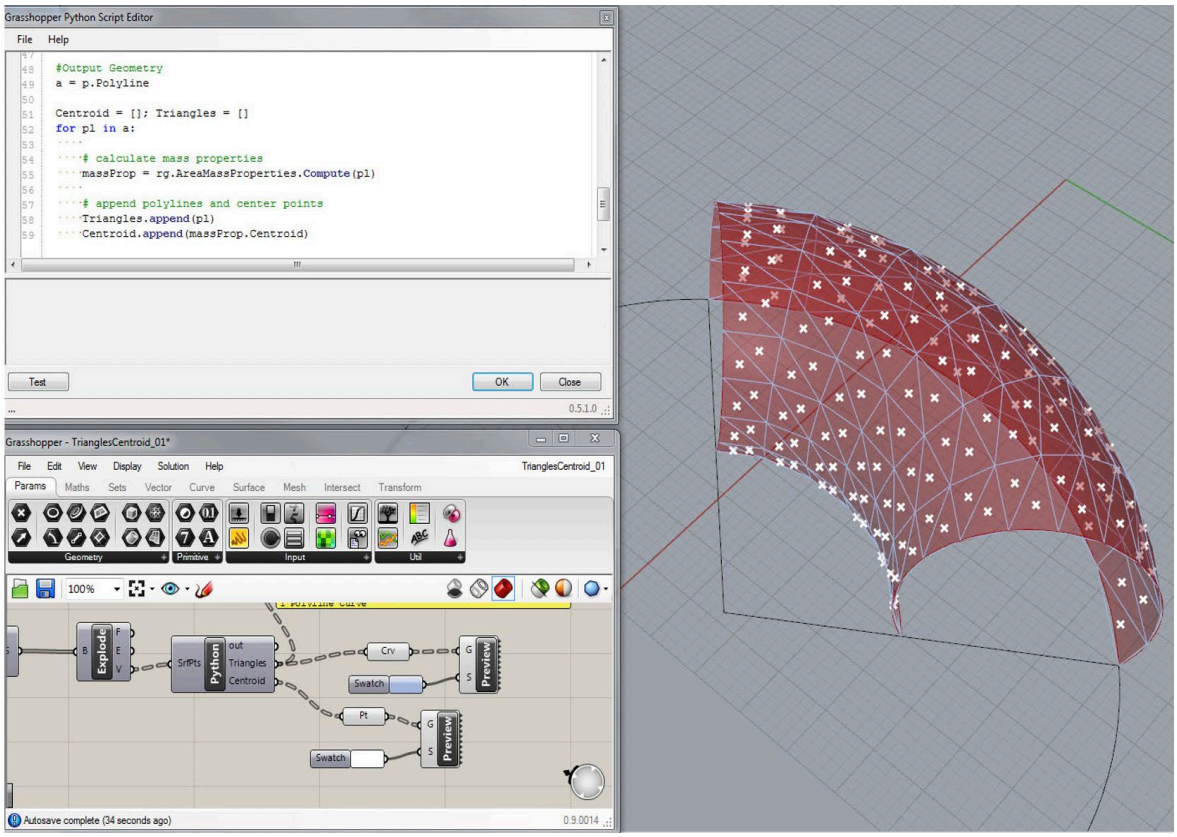
Figura 3 - Programação visual paramétrica



Fonte: Traduzido de SUNG (2010).

Para atenuar essa desvantagem, o Grasshopper permite a criação de pequenos *scripts* em Visual Basic, C# e Python com a utilização de um bloco de código específico. Esses blocos possuem múltiplas entradas e saídas que permitirão a conexão com outros blocos do programa Grasshopper. Apesar da capacidade de criar pequenos *scripts* textuais, esse recurso ainda não elimina totalmente os problemas de escala de um projeto porque tais *scripts* ainda estão vinculados à linguagem visual; a escala e complexidade do projeto podem levar a um alto custo computacional de processamento (FERREIRA; LEITÃO, 2016).

Figura 4 - VPL Grasshopper e Rhinoceros / *sricpt* Phyton



Fonte: Mostapha (2012).

2.1.3 Gramática da Forma

Gramática da Forma é um tipo de sistema generativo baseado em regras que permitem capturar, criar e entender formas. Knight (1994) apresenta que o objetivo inicial era propor um sistema de geração de formas para a pintura e a escultura no qual o artista projetaria suas regras de composição, combinando-as de diferentes maneiras e, então, criaria seu repertório de obras de arte. Desenvolvida por George Stiny e James Gips, a Gramática da Forma foi baseada nos sistemas de produção de Emil Post e nas gramáticas geradoras de Noam Chomsky. Tal como o uso de palavras para formar sentenças, as Gramáticas da Forma funcionam como estruturas para gerar composições formais a partir de formas básicas e regras de associação entre elas.

Estas gramáticas desempenham um papel importante em uma nova geração de ferramentas para análise e criação de formalismos. A estrutura gramatical de

formas permite, além da geração, a representação do conhecimento sobre a lógica formal de um *design,* ou representar o conhecimento sobre a funcionalidade de uma composição. O sistema de gramática da forma foi um dos primeiros sistemas computacionais a viabilizar o desenvolvimento de projetos através da computação de dados sobre as formas. Por meio de uma linguagem visual, tal formalismo possibilitou o emprego da computabilidade (CELANI, 2002; WORTMANN, 2018).

Celani (2002) organiza o desenvolvimento da Gramática da Forma a partir da definição dos seguintes elementos:

1. Vocabulário de formas: conjunto finito de formas primitivas que serão a composição da gramática, podendo ser bi ou tridimensionais;
2. Relações espaciais entre as formas: operações espaciais estabelecidas entre as formas primitivas do vocabulário das formas, como: translação, espelhamento, adição, subtração, rotação, roto-translação e transformação escalar;
3. Regras formais: consistem na combinação das operações e/ou transformações espaciais;
4. Formas primitivas (iniciais): conjunto de formas pertencentes ao vocabulário das formas. Referem-se aos elementos iniciais da construção da concepção do projeto. Através desses elementos conceptivos, pode-se identificar a sucessão e evolução das etapas de composição do projeto com o uso da metodologia da Gramática da Forma.

Celani (2006) demonstra ainda que, com objetivos sintéticos, seus inventores e outros pesquisadores, como Gips (1975), incrementaram o sistema original, acrescentando novas características e novas aplicações da Gramática da Forma:

1. Gramáticas de forma restrita: tipos de gramáticas da forma que podem ser definidos colocando-se mais restrições sobre regras formais permitidas;
2. Gramáticas paralelas: dentro da geração de uma forma paralela, sempre que uma regra formal é utilizada, ela é aplicada, simultaneamente, para todas as partes da forma, em que seja aplicável;

3. Gramáticas matriciais: uma gramática matriz define uma linguagem de dimensões (normalmente, N=2) de matrizes de símbolos;
4. Gramáticas gráficas: uma gramática de rede é definida por um vocabulário de não terminais, um vocabulário de terminais, uma rede inicial e um conjunto de regras de reescrita da rede;
5. Gramáticas de padrões: estas gramáticas são, geralmente, chamadas gramáticas de padrões, ou gramáticas de descrição da imagem;
6. Gramáticas com coordenadas: nas gramáticas com coordenadas, os símbolos terminais e não terminais têm coordenadas que lhes estão associadas;
7. Gramáticas analíticas: estudos analíticos usam um conjunto de projetos existentes para representar a linguagem-*corpus* e inferir as regras da gramática;
8. Gramáticas paramétricas: aquelas em que as regras são parametrizadas de modo que cada regra represente um conjunto de regras;
9. Gramáticas predefinidas: trata-se de um tipo de gramática determinística em que uma mesma regra ou uma mesma sequência de regras é aplicada sucessivamente;
10. Gramáticas com marcadores: o uso de marcadores (*labels*), que são marcas aplicadas às formas para reduzir sua ordem de simetria. Restringem a maneira como as regras podem ser aplicadas, mas permitem maior controle sobre os resultados;
11. Gramáticas da cor: essa variação utiliza cores no lugar de marcadores;
12. Gramáticas discursivas: inclui um conjunto de heurísticas numa gramática da forma.

Entretanto, de forma essencial, para a construção de uma gramática da forma, são necessários três elementos fundamentais: (1) os elementos (ou formas), aos quais são aplicadas as regras; (2) as regras a serem aplicadas, construídas por algum tipo de transformação (euclidiana); e (3) um operador *booleano* (marcadores). As transformações básicas aplicadas às formas consistem em transformações euclidianas, tais como: tradução, rotação, reflexão, escala e identidade, podendo ser feitas em diferentes dimensões e em variados tipos de formas (WORTMANN, 2018; LUDOVICO, 2018).

2.2 Verificação automatizada de projetos

Schumacher (2009) considera que a automação moverá o assunto dos computadores no *design* muito além das narrativas de computação gráficas (CAD / BIM) que dominaram a arquitetura e a engenharia nas últimas duas décadas.

Segundo Campana (2011), a palavra “automação” é proveniente do latim *automatus* e significa “mover-se por si”. De forma prática, a Automação é a aplicação de técnicas computadorizadas ou mecânicas, com o objetivo de tornar um processo mais eficiente, maximizando a produção com menor gasto de energia e gerando maior segurança. Gasto de energia, por sua vez, é compreendido como a aplicação de mão de obra especializada em atividades de baixa geração de valor, gasto de tempo e desperdícios.

O termo Automação começou a ser popularizado pela indústria automobilística americana nas décadas de 1940 e 1950 e foi usado para descrever a mecanização automatizada que estava amadurecendo em todos os tipos de linhas de produção na época. O termo começou a mudar em meados do século 20, com a introdução de ferramentas como máquinas de controle numérico, ou *Computer Numerical Control* (CNC), que eram automaticamente controladas por informações matemáticas codificadas salvas em cartões perfurados. Nos tempos atuais, à medida que o crescimento exponencial do poder da computação se viabiliza, um novo nível de controle de automação digital sobre produção, serviços, e até mídias sociais, continua a surpreender e a se transformar constantemente. Assim, algoritmos, *scripts*, robôs, fabricação digital e novos sistemas de fluxo de trabalho autônomo estão novamente transformando o significado do termo Automação (SCHUMACHER, 2009; HILLER, 2012).

No setor AEC, a verificação automatizada de regras é um processo automatizado de avaliação do projeto em relação aos seus requisitos. Quando processos computacionais mais eficientes ficaram acessíveis, a verificação automatizada de projetos de construção tornou-se um foco de estudo. A redução do tempo e dos custos de avaliações e o aumento da objetividade e da confiabilidade das análises

dos projetos são resultados desta aplicabilidade. Com o surgimento e expansão do BIM, graças ao seu alcance de parametrização, vários sistemas automatizados de verificação de código de construção estão sendo desenvolvidos em projetos reais de edifícios (SCHWABE; KONIG; TEIZER, 2016).

O Solibri Model Checker (SMC), um potente *software* comercial desenvolvido neste âmbito, tem como funcionalidades analisar modelos BIM a partir de um conjunto de regras para identificar e avisar de potenciais problemas, conflitos ou violações que possam existir num determinado modelo paramétrico. O SMC analisa diretamente o modelo BIM, codificado segundo o formato IFC, que retira do modelo apenas os objetos necessários à verificação requerida. O fato deste *software* funcionar por aplicação de regras torna-o versátil, uma vez que descarta a necessidade de conhecimentos de programação para a sua utilização. Efetuada a verificação das regras num determinado modelo, o SMC gera um relatório dos problemas detectados, com o respetivo grau de severidade e com a possibilidade de visualização do problema no próprio modelo virtual (EASTMAN *et al.*, 2009).
Eastman *et al.* (2009a) estruturam o processo de verificação de regras em modelos paramétricos em quatro etapas:

1. Interpretação de regras;
2. Preparação do modelo paramétrico da edificação;
3. Execução de análise das regras;
4. Relatório de verificação de resultados.

Na primeira etapa, os textos de regras naturalmente escritos precisam ser transcritos para uma linguagem de computador interpretável. A segunda etapa trabalha com objetos e seus parâmetros em um modelo 3D. As informações na tabela paramétrica devem ser iguais aos parâmetros dos objetos 3D, para garantir uma rastreabilidade adequada. A etapa três verifica as regras de conformidade. Ambos os valores de retorno – positivo (verdadeiro) e negativo (falso) – são adicionados à estrutura de dados. Um valor de retorno falso detecta um conflito de regras. A forma de um conflito depende da regra que foi associada. Informações de conflito são transferidas para a próxima etapa. A quarta etapa recebe as

informações de conflito da execução da regra e exibe os resultados de duas maneiras diferentes: o relatório textual e o relatório visual. O *SMC* é atualmente o *software* comercial mais robusto na verificação de regras. As regras predefinidas nele incluem desde detecção de conflito físico à comparação de parâmetros dos elementos. Porém, mesmo com a possível edição desses conjuntos de regras predefinidas, essa personalização ainda é limitada (EASTMAN *et al.*, 2009a).

Para Silva e Arantes (2016), em um estudo de caso de um empreendimento, a aplicação da verificação automática de parâmetros constatou redução de 60% no tempo de conferência dos projetos e 22% a mais de não conformidades em relação ao mesmo projeto em uma verificação manual. O trabalho apresenta, portanto, quanto a verificação automatizada de projetos demonstra potencial, mas ressalva que a eficiência do processo está intimamente ligada ao protocolo de desenvolvimento do modelo paramétrico em relação à própria modelagem, parametrização e verificação em cada requisito a ser verificado.

Toda ferramenta de verificação de regras, e, de fato, toda ferramenta de análise BIM, exige que informações paramétricas específicas estejam presentes, e com qualidade, nos modelos virtuais. O enriquecimento semântico de um modelo paramétrico é um processo no qual o conhecimento especializado de uma disciplina técnica oferece a oportunidade de automatizar diversos estágios de verificação do projeto. Este incremento de informações paramétricas abrange, por exemplo, a classificação de objetos de construção, agregação e agrupamento, identificação exclusiva, dedução de objetos ausentes e reconstrução de objetos obstruídos (SACKS; BLOCH, 2018).

2.3 Inteligência artificial

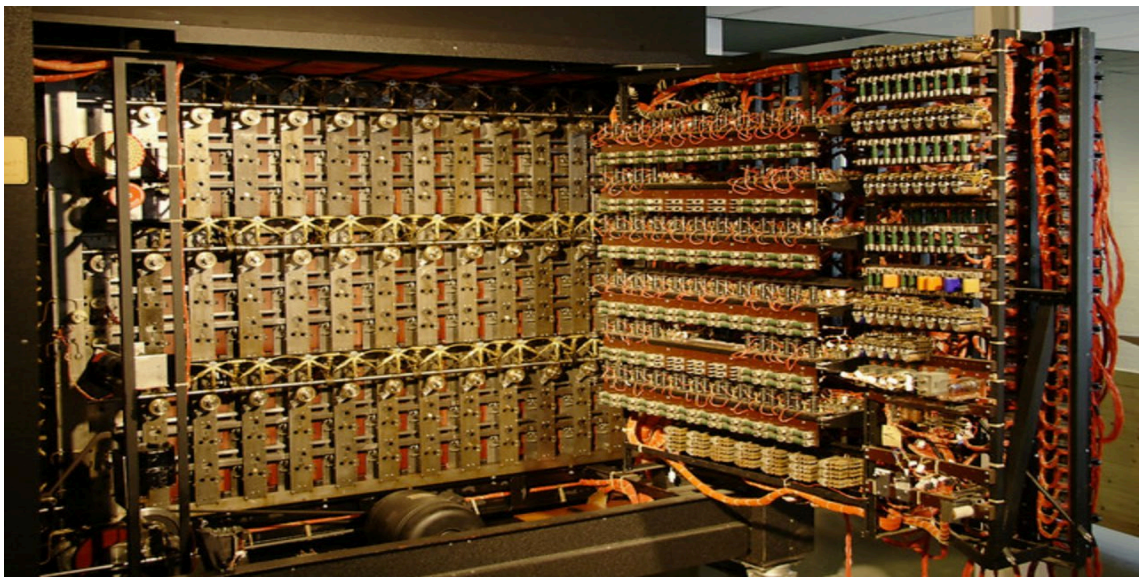
“Can machines think?”. Esta foi uma questão proposta por Alan Turing, quando da publicação do artigo “*Computing machinery and intelligence*” em 1950. Neste momento, o matemático introduziu uma discussão sobre se é justificável chamar um computador de cérebro eletrônico e lançou as bases do que viria a ser o campo da inteligência artificial (TURING, 2009).

Turing foi responsável pela criptoanálise que derrotou a máquina “Enigma” da Alemanha nazista. O cientista, conhecido como o pai da computação, trabalhou neste secreto projeto inglês e, assim, foi também o pioneiro da ciência da computação e da inteligência artificial (HODGES, 2014).

Algoritmo, como define Ashlock (2006), é todo e qualquer procedimento capaz de solucionar um dado problema, através de passos previamente elaborados; e que otimizar significa aperfeiçoar algum processo.

A IA, em seus anos de formação, foi influenciada por ideias de muitas disciplinas. De pessoas que trabalham em engenharia (Norbert Wiener sobre cibernética, que inclui *feedback* e controle), em biologia (W. Ross Ashby e Warren McCulloch no trabalho de Walter Pitts sobre redes neurais em organismos simples), psicologia experimental (Newell e Simon), teoria da comunicação (no trabalho teórico de Claude Shannon), teoria dos jogos (John Von Neumann e Oskar Morgenstern), matemática e estatística (Irving J. Good), lógica e filosofia (nos trabalhos de Alan Turing, Alonzo Church e Carl Hempel) e linguística (com Noam Chomsky sobre gramática). Essas linhas de trabalho deixaram sua marca e continuam a influenciar a IA ainda nos tempos atuais (BUCHANAN, 2005).

Figura 5 - A máquina de *Turing*



Fonte: Acervo *online Bletchley Park* (2013).

Li (2018) aborda uma breve cronologia de entendimentos *versus* expectativas sobre a inteligência artificial:

- gestação da inteligência artificial: 1943 a 1955;
- nascimento da inteligência artificial: 1956;
- entusiasmo precoce, grandes expectativas: 1952 a 1969;
- uma dose de realidade: 1966 a 1973;
- sistemas baseados no conhecimento: a chave do poder?: 1969 a 1979;
- a inteligência artificial se torna uma indústria: 1980 - presente;
- o retorno das redes neurais: 1986 - presente;
- a inteligência artificial adota o método científico: 1987- presente;
- o surgimento de agentes inteligentes: 1995 - presente;
- a disponibilidade do *big data* (conjuntos de dados grandes demais): 2001 - presente).

No histórico sobre AI de Buchanan (2005), somente no último meio século tivemos dispositivos computacionais e linguagens de programação poderosos o suficiente para construir testes experimentais de ideias sobre o que é inteligência. Segundo o autor, o artigo seminal de Turing em 1950 foi um importante ponto de virada na história da IA.

A inteligência artificial é uma área muito ampla, e, segundo Li (2018), mesmo um livro oficial como Russell e Norvig (2003)[1] não fornece uma definição precisa. Para racionalização desta compreensão são listadas a seguir quatro formas de definição da IA:

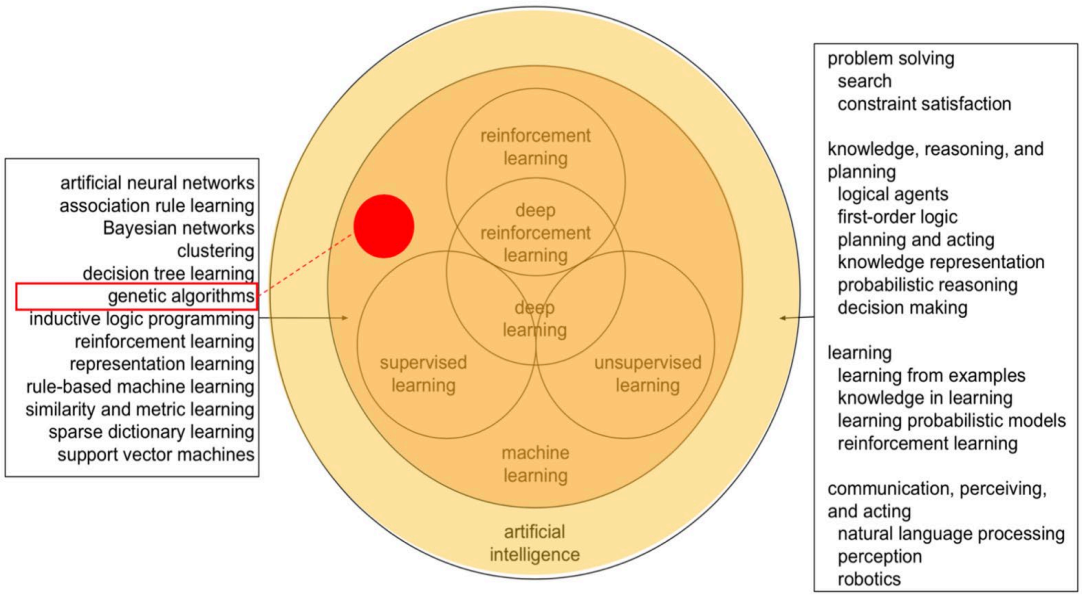
1. Agindo humanamente: seguindo uma abordagem chamada Teste de Turing, na qual um computador passa no teste se um interrogador humano não pode dizer se as respostas a alguma pergunta são de um computador ou de um ser humano.

[1] Conforme a editora, a primeira edição de “Inteligência artificial - uma abordagem moderna” tornou-se um clássico na literatura sobre IA. O livro foi adotado por mais de 600 universidades de 60 países e foi elogiado como a síntese definitiva desse campo.

2. Pensando humanamente: uma abordagem da modelagem cognitiva, na qual se faz os computadores pensarem em atividades que associamos ao pensamento humano, como tomada de decisão, resolução de problemas, aprendizado.
3. Pensando racionalmente: abordagem da "lei do pensamento", quando as faculdades mentais, através do uso de modelos computacionais, possibilitam perceber, raciocinar e agir.
4. Agindo racionalmente: segue a abordagem do agente racional ou Inteligência Computacional, onde a IA está preocupada com o comportamento inteligente em artefatos, sendo esta mais favorável à ciência do que aquelas abordagens baseadas no comportamento ou pensamento humano.

Domingos (2017) e Li (2018) compreendem que a IA atualmente possui diversas vertentes e aplicabilidades distintas e, assim, algoritmos e modelos matemáticos variam conforme a natureza do problema a ser resolvido computacionalmente. Em notável expansão e aplicabilidade atualmente, o aprendizado de máquinas (*machine learning*), por exemplo, é um subcampo da IA voltado ao desenvolvimento de algoritmos e técnicas que permitam ao computador aprender.

Figura 6 - Diagrama de inteligência artificial e seus subcampos



Fonte: LI (2018).

Em *machine learnin*g, computadores são programados para aprender com a experiência passada. Para isso, utilizam um princípio de inferência denominado indução, obtendo conclusões genéricas a partir de um conjunto particular de exemplos. Assim, conforme Faceli *et al.* (2011), algoritmos de *machine learning* aprendem a induzir uma função ou hipótese capaz de resolver um problema a partir de dados que representam instâncias do problema a ser resolvido.

> Quando um algoritmo de *machine learning* está aprendendo a partir de um conjunto de dados de treinamento, ele está procurando uma hipótese, no espaço de possíveis hipóteses, capaz de descrever as relações entre os objetos e que melhor se ajuste aos dados de treinamento. Cada algoritmo utiliza uma forma ou representação para descrever a hipótese induzida. Por exemplo, redes neurais artificiais representam uma hipótese por um conjunto de valores reais, associados aos pesos das conexões da rede. Árvores de decisão utilizam uma estrutura de árvore em que cada nó interno é representado por uma pergunta referente ao valor de um atributo e cada nó externo está associado a uma classe. A representação utilizada define a preferência ou viés (*bias*) de representação do algoritmo e pode restringir o conjunto de hipóteses que podem ser induzidas pelo algoritmo. Além do viés de representação, os algoritmos de *machine learning* possuem também um viés de busca. O viés de busca de um algoritmo é a forma como o algoritmo busca a hipótese que melhor se ajusta aos dados de treinamento. Ele define como as hipóteses são pesquisadas no espaço de hipóteses (FACELI *et al.* 2011, p.5).

Algoritmos de *machine learning* podem ser organizados de acordo com diferentes critérios, como o paradigma de aprendizado a ser adotado para lidar com uma tarefa. Sobre esse critério, Arel *et al.* (2010) e Faceli *et al.* (2011) dividem as tarefas de aprendizado em:

1. Preditivas: em tarefas de previsão, a meta é encontrar uma função (modelo ou hipótese) a partir dos dados de treinamento que possa ser utilizada para prever um valor que caracterize um novo exemplo, com base nos valores de seus atributos de entrada. Para tanto, cada objeto do conjunto de treinamento deve possuir atributos de entrada e de saída. Os algoritmos de modelos preditivos seguem o paradigma de aprendizado supervisionado, onde há a

simulação da presença de um “supervisor externo", que conhece a saída (rótulo) desejada para cada exemplo (conjunto de valores para os atributos de entrada). Com isso, o supervisor externo pode avaliar a capacidade da hipótese induzida de predizer o valor de saída para novos exemplos.

2. Descritivas: em tarefas de descrição, o objetivo é explorar ou descrever um conjunto de dados. Nessas tarefas, os algoritmos de *machine learning* não fazem uso do atributo de saída. Por esta razão, seguem o paradigma de aprendizado não supervisionado. Uma tarefa descritiva de agrupamento de dados, por exemplo, tem por meta encontrar grupos de objetos semelhantes no conjunto de dados. Outra tarefa descritiva é encontrar regras de associação que relacionam um grupo de atributos a outro grupo de atributos.

Buchanan (2005) lembra que se os primeiros programas eram necessariamente limitados em escopo pelo tamanho e velocidade da memória e processadores e pela relativa falta de jeito dos sistemas operacionais e linguagens anteriores, atualmente as aplicabilidades e efetividade dos algoritmos se multiplicaram.

Sacks e Bloch (2018) exemplificam que as técnicas de inteligência artificial e *machine learning* têm sido mais aplicadas por pesquisadores a vários problemas na indústria de AEC: uma rede neural artificial utilizada para prever o consumo de energia em edifícios (os recursos considerados foram orientação do edifício, propriedades do isolamento etc.); classificar e rotular superfícies obtidas a partir de uma varredura a *laser* (classificaram assim os principais objetos construtivos de um edifício, como paredes, tetos e pisos, e separá-los de outros objetos obtidos a partir da digitalização que são considerados desorganizados).

Brígitte e Ruschel (2018) ponderam que algoritmos matemáticos têm se tornado particularmente importantes por servirem como base para abordagens e estratégias paramétricas também na indústria AEC.

2.4 Computação evolucionária

A Computação Evolucionária também é compreendida como uma área da inteligência computacional ou inteligência artificial destinada à resolução de problemas de otimização através do uso de processos iterativos. Tem como objetivo desenvolver, avaliar e aplicar técnicas na criação de algoritmos inteligentes (ASHLOCK, 2006).

A terminologia contemporânea denota que os algoritmos envolvidos na computação e programação evolucionária são denominados Algoritmos Evolucionários, que empregam princípios semelhantes aos da seleção natural para busca de soluções otimizadas (EIBEN; SMITH, 2008).

Para Lopes e Takahashi (2011), projetos de engenharia são um campo no qual se requer tanto o conhecimento de cada especialidade da engenharia como também o uso de técnicas capazes de tratar melhores soluções possíveis para um mesmo projeto que mereceriam ser chamadas de ótimas, ou de alto desempenho, econômicas e confiáveis. Para encontrar tais soluções são necessárias técnicas de otimização. Para os autores, devido ao aumento da complexidade dos sistemas a serem projetados, à medida em que avançam as tecnologias, torna-se cada vez mais importante o uso da otimização através de algoritmos.

Algoritmos de otimização resolvem problemas em que a pesquisa exaustiva seria inviável. Entende-se como pesquisa exaustiva quando, para um determinado problema, são testadas todas as diferentes possibilidades e então escolhida a melhor. Porém, determinados problemas gerariam um número tão grande de alternativas, que estaria além dos limites, não apenas de um humano avaliá-las, mas também mesmo computadores poderosos seriam incapazes de gerá-las. Ao resolver problemas de otimização onde é impraticável testar cada solução possível, pode-se contar com soluções aproximadas, garantindo que se alcance uma solução razoável, mesmo não sendo a melhor possível. Na ciência da computação, esses métodos são chamados de "heurísticas". Embora ágeis e eficazes, as estratégias heurísticas devem ser desenvolvidas especificamente para cada estratégia de

projeto e não são generalizáveis para diferentes problemas. Entretanto, nem sempre os problemas podem contar com estratégias de métodos diretos – aqueles que podem ser derivados deterministicamente seguindo um conjunto fixo de regras. Quando pouco se sabe sobre a relação dos parâmetros de entrada e as funções de otimização, uma alternativa é testar soluções aleatórias e considerar a melhor solução encontrada como a ótima. Essas abordagens puramente aleatórias são chamadas de “métodos de Monte Carlo”. Mas uma estratégia puramente aleatória dificilmente encontraria uma boa solução em um período de tempo razoável. Para permitir um resultado satisfatório de busca, é importante evitar que ela se direcione e termine em uma única direção de soluções (ótimo local). Para isso, são necessárias muitas iterações do algoritmo, iniciando em pontos diferentes, a cada iteração (uma abordagem chamada “descida de gradiente estocástica”). Atualmente, algoritmos estocásticos baseados neste princípio são usados em diversas aplicações como otimizações, criptografia e *machine learning*. A classe de algoritmos de otimização que se baseia em princípios estocásticos é chamada de “metaheurística”. O termo refere-se, portanto, ao conjunto de regras que orientam o processo de otimização de cada algoritmo. Assim, diferentemente dos métodos heurísticos, as abordagens metaheurísticas são generalizáveis para qualquer problema de otimização. Embora sejam uma abordagem relativamente nova para resolver problemas de otimização, permite a escolha de muitos tipos de algoritmos. Muitos desses algoritmos são inspirados por processos encontrados na natureza, com sua capacidade em resolver problemas complexos por meio de uma abordagem iterativa e estocástica (EIBEN; SMITH, 2008; BORENSTEIN; MORAGLIO, 2014; NAGY, 2017b; NAGY, 2017d).

Eiben e Smith (2008) discorrem que a metáfora fundamental da computação evolucionária relaciona essa poderosa evolução natural a um estilo particular de solução de problemas: o de tentativa e erro. Essa ideia de aplicar os princípios darwinianos à solução automatizada de problemas data da década de 1940, muito antes do avanço dos computadores. E, já em 1948, Turing propôs “pesquisa genética ou evolutiva”, e, em 1962, Bremermann já havia realizado experiências computacionais sobre “otimização através da evolução e recombinação”. Da década de 1960 a 1990, quatro implementações diferentes da ideia básica foram

desenvolvidas: programação evolutiva, algoritmo genético, estratégias de evolução e programação genética. Depois, estes diferentes dialetos de uma mesma tecnologia passaram a ser conhecidos como Computação Evolucionária. A terminologia contemporânea denota todo o campo pela Computação Evolucionária, onde os algoritmos envolvidos são denominados algoritmos evolucionários, e considera a programação evolutiva, estratégias de evolução, algoritmos genéticos e programação genética como subáreas variantes.

Os principais algoritmos evolucionários (ou evolutivos) são os Algoritmos Genéticos (AG), a Programação Genética (GP), a Otimização por Colônia de Formigas (ACO) e a Otimização por Enxame de Partículas (PSO). Estas três últimas conhecidas como Inteligência de Enxames (*Swarm Intelligence*) (LOPES; TAKAHASHI, 2011).

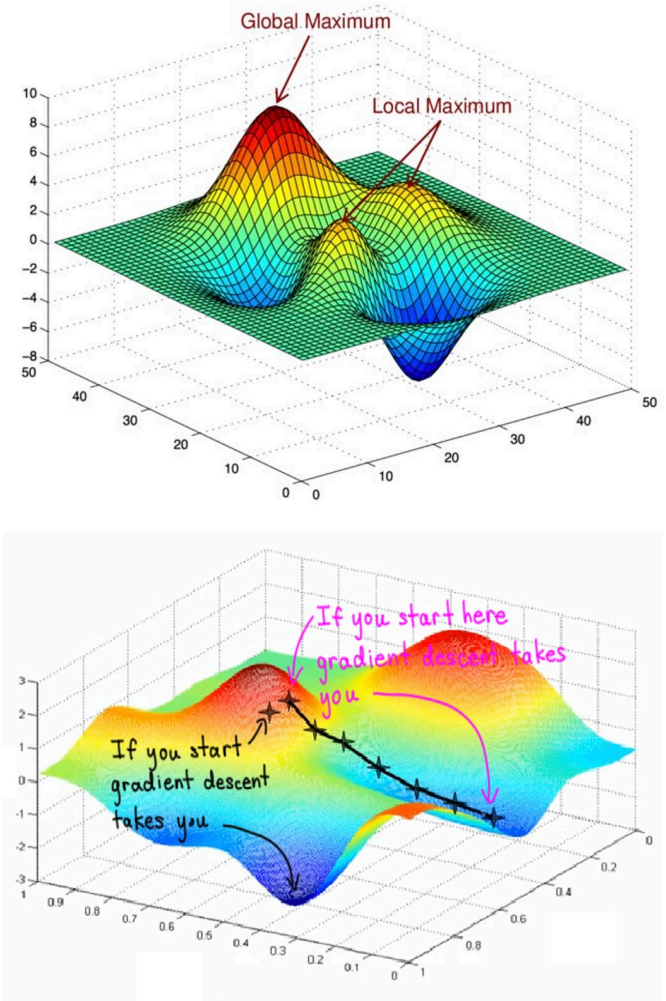
A escolha do método, conforme defende Davis (1991), depende de uma série de características do problema a ser otimizado, principalmente do comportamento da função que o representa.

2.4.1 Algoritmos evolutivos genéticos (AG)

Os Algoritmos Genéticos (AG) são inspirados na Teoria da Origem das Espécies de Charles Darwin, onde uma população desenvolve-se durante várias gerações, segundo os princípios da sobrevivência do mais apto. Os AG foram propostos para desenvolver soluções para problemas do mundo real, como problemas de busca e otimização. Entretanto, a teoria aponta que não há garantia de se encontrar a melhor solução do problema, ou seja, o ótimo global, e sim a melhor solução possível, podendo ser um ótimo local ou, eventualmente, o global, pois não é feita a varredura completa do espaço de busca. Esta técnica de resolução de problemas foi desenvolvida por John Henry Holland, em 1975, e os principais objetivos de sua pesquisa eram: (a) abstrair e explicar rigorosamente os processos adaptativos dos sistemas naturais; (b) desenvolver programas de sistemas artificiais capazes de simular a robustez dos mecanismos dos sistemas naturais. Sendo que esta evolução ocorre através da utilização de um tipo de seleção natural em conjunto com outros operadores genéticos, tais como: mutação (onde um gene é substituído

por um alelo de forma aleatória), e recombinação ou cruzamento (onde os indivíduos escolhidos trocam pedaços de seus cromossomos e, assim, geram novas soluções). Um AG busca produzir indivíduos melhores com base na combinação dos melhores existentes na população; e, por meio da estratégia da sobrevivência do mais apto, elimina os menos aptos. Não são produzidas apenas boas soluções, mas sempre soluções melhores. Isto ocorre devido ao fato de serem combinadas as melhores características dos pais, na produção de filhos superiores. Assim, os AG são algoritmos de busca baseados nos mecanismos de seleção natural e genética, que têm a capacidade de evoluir uma população de indivíduos. (DAVIS, 1991; LOPES; TAKAHASHI, 2011).

Figura 7 - Ótimo global e ótimo local



Fonte: JIN (2015).

Lopes e Takahashi (2011) apresentam a estrutura proposta por Holland:

Figura 8 - Estrutura básica de um Algoritmo Genético

```
1. Início;
2. Gerar população inicial;
3. Avaliar cada indivíduo da população inicial;
4. Repetir até m gerações
// Etapa de reprodução
   4.1. Repetir até o número de descendentes ser igual à quantidade
        desejada (% de cruzamento)
   4.1.1. Selecionar dois indivíduos conforme o método de seleção;
   4.1.2. Realizar a operação de cruzamento nos indivíduos selecionados no
          passo anterior;
   4.1.3. Realizar a operação de mutação nos descendentes gerados no passo
          anterior;
   4.1.4. Avaliar os descendentes;
   4.2. Substituir os indivíduos da população pelos novos indivíduos
        gerados na etapa de reprodução;
5. O melhor indivíduo da população é a solução do problema;
6. Fim.
```

Fonte: Lopes e Takahashi (2011).

A técnica de Holland também é descrita por Davis (1991), conforme a sequência:

1. Iniciar uma população aleatoriamente de tamanho N (ou seja, criar N cromossomos com conteúdo aleatório que será a população inicial);
2. Aplicar a função de avaliação (*fitness*) em cada um desses cromossomos, obtendo uma classificação dos mais adaptados ao problema;
3. Fazer o cruzamento dos cromossomos, levando-se em consideração que os mais adaptados devem ter maiores chances de reprodução. Esse procedimento poderá gerar novos cromossomos;
4. Aplicar mutação a uma pequena porcentagem dos cromossomos criados. A mutação pode proporcionar o aparecimento de boas surpresas;
5. Eliminar os cromossomos da população antiga, de forma que os novos cromossomos gerados possam ser inseridos sem alterar o tamanho da população inicial;
6. Aplicar a função de avaliação e inserir os melhores selecionados na população anterior, gerando uma nova população;
7. Se a população de cromossomos atual representar o resultado esperado ou se a quantidade de gerações máxima definida for atingida, parar. Caso contrário, voltar à etapa 3.

O tamanho N da população inicial é uma variável do algoritmo e seu valor depende da complexidade do problema. Valores típicos encontram-se entre 20 e 200 indivíduos. Ao final do algoritmo acima espera-se que a população de cromossomos gerada seja mais bem adaptada à função de avaliação, sendo, dessa forma, a que melhor represente o resultado do problema (DAVIS, 1991).

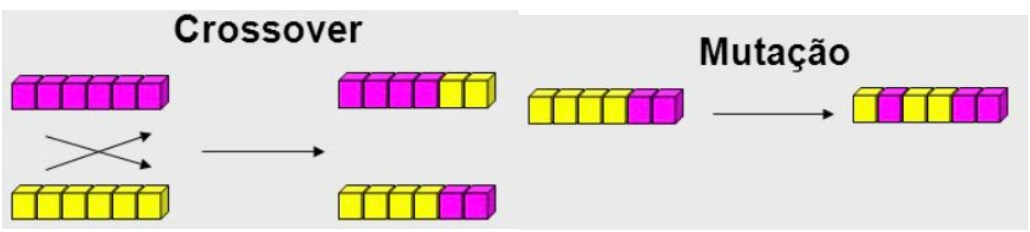
Neto (2011) sintetiza as condições do funcionamento de um AG:

- **Codificação (geração dos cromossomos)**: deverá ser de tal forma que represente uma solução do problema e deverá possuir a característica de poder ser dividida em partes menores, chamadas de genes. Pode-se codificar o cromossomo como sendo um conjunto de números reais, ou um conjunto de números inteiros, ou até mesmo, e mais comumente, um conjunto de *bits*.
- **Função de avaliação (*fitness*)**: basicamente um modo de avaliar o quão distante um dado cromossomo está da solução ótima. É ela quem faz a ligação entre o problema real e o Algoritmo Genético, uma vez que os cruzamentos são feitos com os indivíduos mais bem adaptados. Sem a função de avaliação seria impossível determinar se um problema está evoluindo para a solução ótima ou não.
- **Seleção:** forma como, fazendo uso da função de avaliação, serão classificados e selecionados os cromossomos. Os operadores de seleção envolvem algoritmos que utilizam métodos determinísticos ou probabilísticos como o elitismo, roleta (*roulette wheel*) e torneio (*ranking*).
  - **Elitismo:** consiste em um caso particular de seleção na qual um ou mais dos melhores indivíduos da população é sempre mantido e nenhum dos piores indivíduos é selecionado.
  - **Roleta:** atribui, a cada indivíduo de uma população, uma probabilidade de passar para a próxima geração. A probabilidade é proporcional a seu *fitness* normalizado, ou seja, o *fitness* medido em relação ao somatório do *fitness* de todos os indivíduos da população.
  - **Torneio:** para se selecionar *N* indivíduos, realizam-se *N* torneios envolvendo *q* indivíduos em cada torneio, escolhidos sem levar em conta o *fitness* e com

reposição (um indivíduo pode aparecer mais de uma vez num mesmo torneio). Vence cada torneio aquele que apresentar o maior *fitness*.

- **Reprodução:** processo através do qual se combinam dois cromossomos (pais) para formar filhos a partir deles. Tais filhos herdam as características dos pais através da herança de material genético (genes). Esse processo é conhecido como recombinação. Além da recombinação, está envolvida, no processo de reprodução, a mutação. Na mutação, altera-se aleatoriamente o material genético de uma pequena parcela dos filhos gerados.
  - **Recombinação (*crossover*):** processo de troca de material genético entre dois cromossomos. Pode-se definir aqui a probabilidade de recombinação, que indica a probabilidade de dois indivíduos escolhidos no processo de seleção virem a se reproduzir. Quebram-se os dois cromossomos envolvidos (em um ou mais pontos), juntando-se a primeira parte do cromossomo A com a segunda parte do cromossomo B e a primeira parte do cromossomo B com a segunda parte do cromossomo A, gerando, assim, dois filhos.
  - **Mutação:** utilizada para que sejam gerados novos cromossomos distintos dos pais. É de grande importância, pois a população inicial é gerada aleatoriamente e as novas gerações são frutos da recombinação da população inicial. Se nenhum cromossomo da população inicial possuir carga genética do ponto ótimo, passarão muitas gerações até que surja um cromossomo que possua as características desejadas. Na prática, escolhe-se, aleatoriamente, um gene de alguns cromossomos e troca-se o seu valor por outro também aleatório. Se os cromossomos forem representados por cadeias de *bits*, isto significa trocar o valor 0 por 1 e vice-versa (DAVIS, 1991; FACELI *et al., 2011;* LOPES; TAKAHASHI, 2011).

Figura 9 - Operações genéticas: *crossover* e mutação



Fonte: Adaptado de Lopes e Takahashi (2011).

A computação evolucionária usa um processo para criar uma saída chamada mapeamento de genótipo para fenótipo, uma terminologia também adaptada da biologia. Um genótipo é uma codificação sobre a qual agem os operadores evolutivos de mutação e recombinação. Um exemplo biológico de um genótipo seria o DNA humano; e o fenótipo uma característica observável de um indivíduo – como a altura, por exemplo. A avaliação de uma aptidão é realizada na observação do fenótipo (EIBEN; SMITH, 2008).

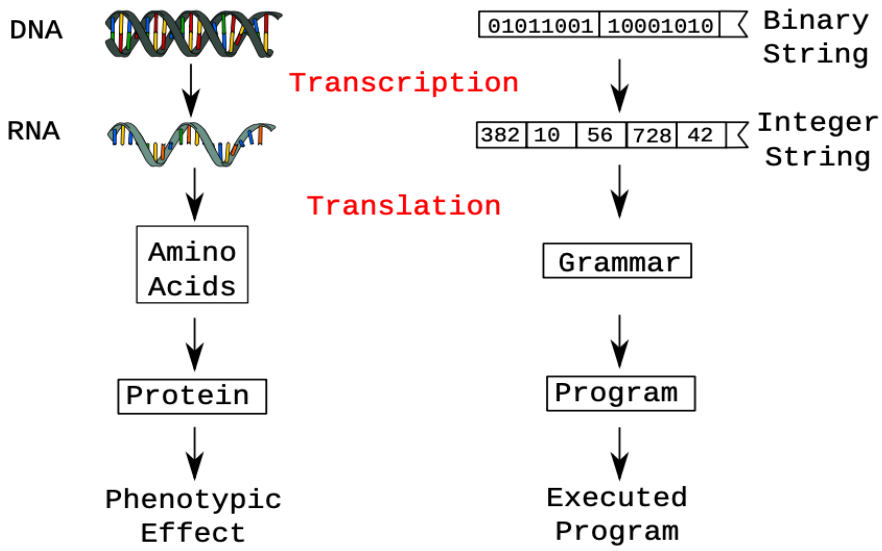
Figura 10 - Genótipo e fenótipo

Fonte: Valeri Natole (2019).

Na CE este conceito biológico foi introduzido como um método para separar o espaço de busca e solução e também como uma metáfora para descrever as representações e processos de mapeamento, mostrados na Figura 11. A cadeia numérica é análoga à dupla hélice do DNA – ambas são responsáveis por definir as características do fenótipo. Assim, a computação evolucionária gera *strings* usando uma lista de valores numéricos na representação genotípica que podem ser armazenados como valores binários ou inteiros. As *strings* de saída são a forma fenotípica do indivíduo. Os fenótipos são avaliados pela função de aptidão do algoritmo (BYRNE, 2012).

A aplicação de técnicas evolutivas ocorre, muitas vezes, a uma classe particular de problemas: a otimização multiobjetivo. Problemas multiobjetivos (MOPs) são aqueles onde a qualidade de uma solução é definida por seu desempenho em relação a mais de um objetivo, possivelmente conflitantes entre si (EIBEN; SMITH, 2008). Na prática, porém, verifica-se que, em muitas situações, um problema MOP é transformado em uma função de objetivo único (*single-objective optimization*), a fim de tornar a otimização computacionalmente mais viável (CHANG, 2015).

Figura 11 - Mapeamento de genótipo para fenótipo na CE

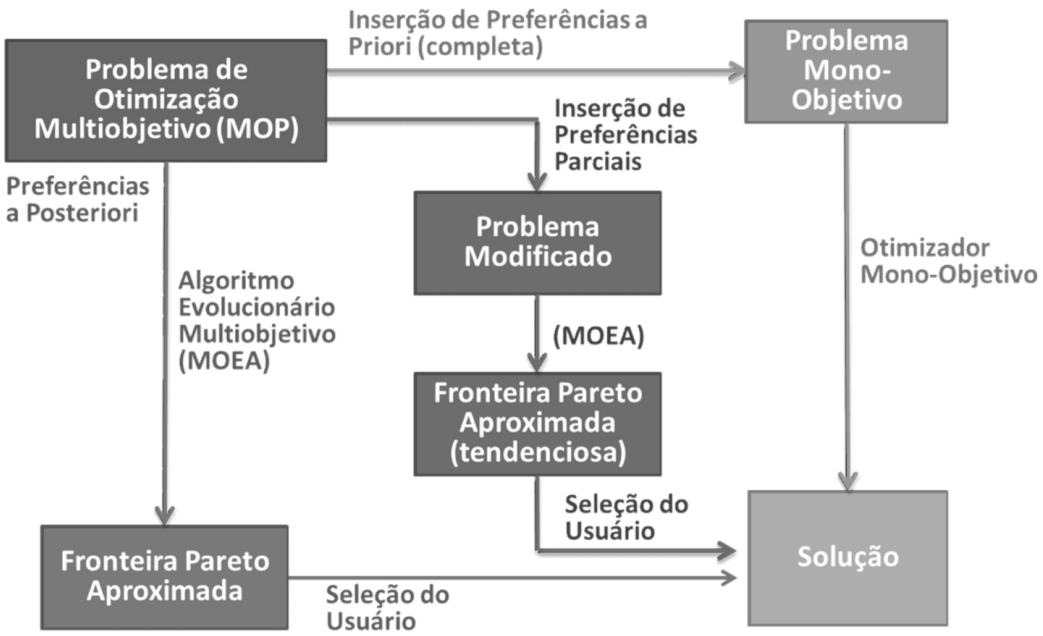


Fonte: Byrne (2012).

De acordo com Cuadros *et al.* (2013), em um cenário que envolva um problema de otimização com múltiplos objetivos igualmente importantes, as técnicas de solução de funções MOP podem ser divididas em três categorias principais. Estas categorias são baseadas na preferência do procedimento de aquisição da informação e no uso desta informação nos processos de análise de decisão. São elas: (i) Articulação *a priori*; (ii) Articulação progressiva; (iii) Articulação *a posteriori*. A "articulação *a priori*" é a forma mais utilizada, o que significa que diferentes objetivos são convertidos em um único item antes de realizar a operação de otimização real.

Existem inúmeras maneiras de categorizar os métodos de solução, como aplicação de escalas *versus* métodos de Pareto, "articulação *a priori*" *versus* "*a posteriori*", e assim por diante. Embora muitas técnicas de solução tenham sido desenvolvidas nas últimas décadas, nenhum desses métodos é perfeito, e a seleção dentre eles dependerá dos requisitos do projeto em particular. A "articulação *a priori*" implica que o projetista indique a importância relativa das funções objetivo ou metas desejadas antes de executar o algoritmo de otimização. Com as preferências especificadas, o multiobjectivo é convertido em um único problema de otimização, levando a uma única otimização.

Figura 12 - Otimização evolutiva e tomada de decisão



Fonte: De Vasconcelos (2016).

No entanto, valores de funções diferentes podem ter unidades diferentes e/ou limites significativamente diferentes, tornando as comparações difíceis. Assim, geralmente, é necessário transformar as funções objetivo de modo que todas tenham parâmetros semelhantes. Embora existam diferentes abordagens propostas para tal propósito, uma das mais simples é normalizar as funções objetivo individuais por seus respectivos valores de função absoluta do projeto em questão (CHANG, 2015). A Figura 13, a seguir, exemplifica uma função normalizada em uma "articulação *a priori*".

Figura 13 - Função normalizada em "articulação *a priori*"

$$f_i^{\text{norm}}(\mathbf{x}) = \frac{f_i(\mathbf{x})}{|f_i(\mathbf{x}^0)|}$$

Fonte: Chang (2015).

2.5 *Design* evolucionário

O *Design* Evolucionário tem suas raízes na ciência da computação e na biologia evolucionária. É um ramo da Computação Evolucionária, estende e combina *softwares* de CAD e análises, emprestando de ideias da evolução natural. Bentley (1999) divide os tipos variados de *design* evolucionário em quatro categorias principais: 1) otimização do *design* evolucionário; 2) *design* evolucionário criativo; 3) arte evolucionária; e 4) formas de vida artificiais evolucionárias.

Jones (2009) avalia que o desenvolvimento de projetos em computadores agora permite empregar computadores em todas as etapas do processo de *design*. Não sendo mais o *design* assistido por computador, mas tornando o *design* produzido pelo computador.

Como os problemas de *design* oferecem um número inesgotável de soluções, o processo de *design* pode não ter um fim identificável. O trabalho dos *designers* nunca é realmente concluído e, provavelmente, sempre é possível fazer melhor, conforme afirma Jo (1998).

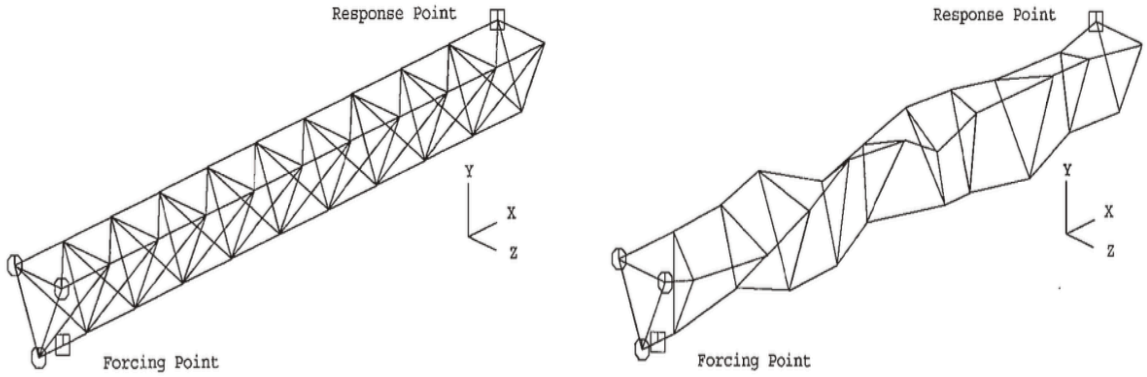
A capacidade de um algoritmo evolutivo de otimizar uma solução foi aplicada aos problemas de *design* desde o início do *design* computacional. Byrne (2012) relata que Rechenberg demonstrou uma das primeiras aplicações práticas da computação evolucionária. Posteriormente, a abordagem acabou se desenvolvendo no campo das estratégias evolutivas e a aplicabilidade demonstrou que o computador poderia desenvolver um projeto conceitual e depois permitir que o refinasse ainda mais.

Calixto e Celani (2015) apresentam uma revisão da literatura sobre a otimização do planejamento espacial usando a abordagem de algoritmos evolutivos de 1992 a 2014. Segundos os referidos autores, o uso desses algoritmos em problemas de planejamento espacial tem sido explorado desde o início dos anos 1990, e ainda não foi totalmente esgotado. O Anexo A apresenta a lista destes artigos revisados pelos autores.

Como é possível verificar na abordagem, embora estudos sobre a aplicabilidade dos algoritmos evolucionários em projetos de *design* já sejam amplos, não se verifica a relação do tema com a avaliação automatizada de projetos, tampouco em relação à geração de projetos paramétricos BIM.

As aplicações evolucionárias na otimização de soluções de engenharia podem ser ilustradas com o caso de uma longarina de suporte de antena parabólica: uma construção em forma de escada conecta o corpo do satélite com o prato necessário para a comunicação. Como requisitos espera-se que esta barreira seja estável e, em particular, resistente a vibrações, pois não há elementos que possam amortecer as vibrações que quebrariam toda a construção. A imagem da Figura 14 representa uma otimização desta construção usando um algoritmo evolutivo, resultando em uma estrutura 20.000% melhor do que as formas tradicionais. Para o senso comum, pode parecer uma solução muito estranha: não possui nenhuma simetria e não há nenhuma lógica de *design* intuitiva visível, parecendo uma forma aleatória. De fato, um desenho aleatório, desenhado sem inteligência, mas evoluindo através de várias gerações consecutivas de soluções de melhoria. Um *design* puramente direcionado pela qualidade e que, assim, pode chegar a soluções que estão fora do âmbito do pensamento humano, muitas vezes, limitado por convenções, considerações estéticas ou preferências infundadas (EIBEN; SMITH, 2008).

Figura 14 - Design de uma longarina otimizada por processo evolucionário



Fonte: Eiben e Smith (2008).

Eiben e Smith (2008) trazem, ainda, que o *design* evolucionário, muitas vezes, anda de mãos dadas também com a engenharia reversa: uma vez que uma solução comprovadamente superior é desenvolvida, ela pode ser analisada e explicada pela engenharia tradicional. Isso pode levar a conhecimentos generalizáveis com a formulação de novas leis, teorias ou princípios de *design* aplicáveis a uma variedade de outros problemas de tipo semelhante.

De acordo com Guimarães (2008), em projetos assistidos por computador, tem-se que:

> O processo começa com o estabelecimento de especificações e requisitos, o que leva à definição de um protótipo, um primeiro modelo de projeto para o problema em questão. Essas etapas iniciais dependem do conhecimento e intervenção de um *designer* humano. O próximo passo é gerar o modelo geométrico computacional do protótipo, cuja descrição depende do método de análise a ser utilizado. O principal objetivo do protótipo (e seu modelo computacional associado) é fornecer um espaço de pesquisa parametrizado e um conjunto de funções de objetivo e restrição para o processo de otimização. O objetivo do processo de otimização é encontrar a solução, no espaço de pesquisa fornecido, que move o projeto atual (protótipo atual) para o protótipo mais próximo possível que atenda aos requisitos. Finalmente, se o protótipo otimizado atender às especificações e requisitos definidos pelo *designer*, o processo de *design* será interrompido e a solução final será alcançada (GUIMARÃES, 2008, p.32).

Ocorre que os problemas de *design*, na maioria das vezes, são definidos por muitos objetivos diferentes, que podem estar relacionados entre si de maneiras complexas e não intuitivas. Mesmo para um problema de otimização multiobjetivo trivial, é improvável que exista uma única solução que otimize simultaneamente cada objetivo. Pesquisadores estudam MOPs sob diferentes pontos de vista, com diferentes abordagens de solução e objetivos. O objetivo pode ser encontrar um conjunto representativo de soluções ótimas de Pareto, satisfazer os diferentes objetivos e/ou encontrar uma única solução que satisfaça as preferências definidas pelo projetista (CHANG, 2015).

3 MÉTODO DA PESQUISA

O método de pesquisa adotado neste estudo consiste na aplicação da *Design Science Research* (DSR), também conhecida como *Constructive Research* ou Pesquisa Construtiva.

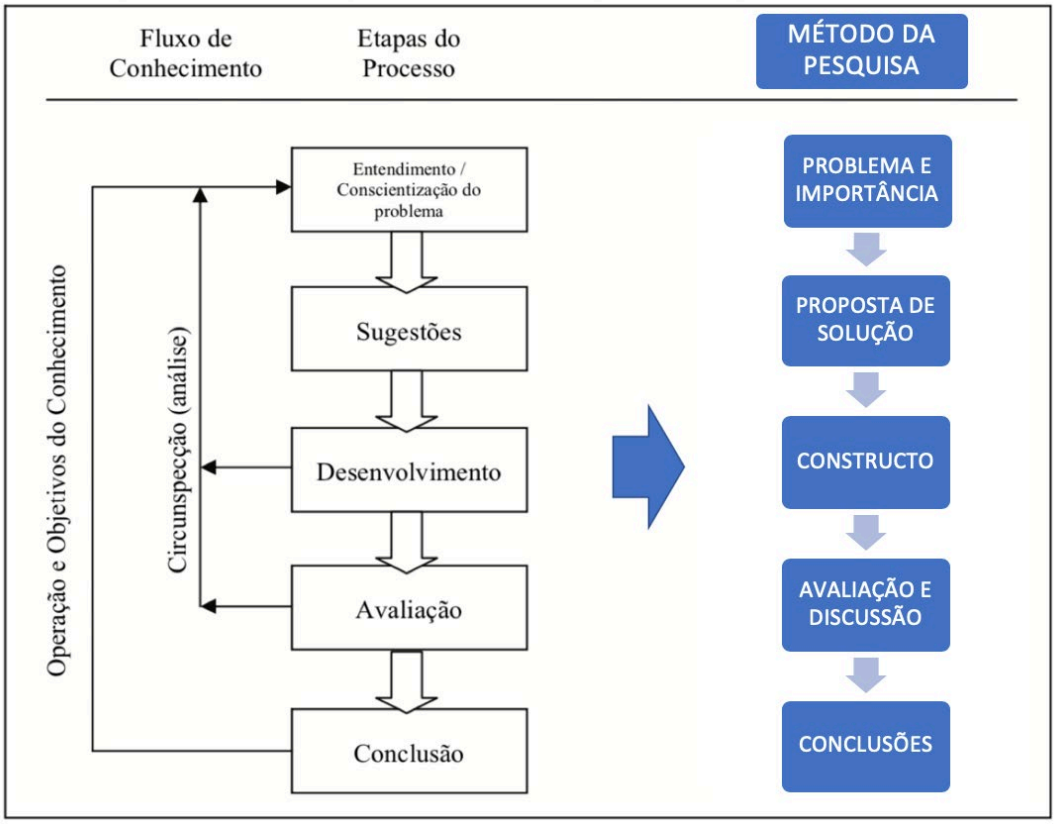
A Pesquisa Construtiva tem como finalidade conceber um conhecimento, e não somente explicá-lo ou aplicá-lo. Simon (1996) afirma que a ciência tradicional não pode ser considerada a única fonte de construção do conhecimento. A análise decorre da justificativa de que, embora também importantes, as ciências tradicionais têm como objetivos centrais descrever e explorar, não sendo suficientes para a criação de elementos. Apesar da comparação, o autor esclarece que as ciências tradicionais e a DSR não são opostas, ao contrário, se complementam e apenas possuem sentidos distintos. Destaca ainda que um artefato é a interface entre o mundo natural e o artificial.

Assim, a DSR muda o estado do mundo por meio da introdução de novos artefatos. Usando criação, análise, reflexão e abstração, o método cria conhecimento na forma de constructos, técnicas e métodos, modelos, e/ou teoria bem desenvolvida para a solução de problemas reais (VAISHNAVI; KUECHLER, 2004).

Embora lide com um problema real, Lukka (2003) aborda que a solução proposta pela Pesquisa Construtiva deve permitir uma generalização para um determinado “Tipo de Problema”. Neste sentido, a DSR é principalmente utilizada em áreas com demandas de intervenções práticas, como engenharias e arquitetura. Seja o resultado da Pesquisa Construtiva positivo ou negativo, ela é relevante pelo legado da avaliação empírica na medida em que se torna embasamento de futuras pesquisas no tema.

Vaishnavi e Kuechler (2004) estabelecem uma sequência de etapas para atingir os objetivos estabelecidos no desenvolvimento e a aplicação da metodologia *Design Science Research,* compreendendo as etapas da Figura 15:

Figura 15 - Mapa do processo DSR



Fonte: adaptado de Vaishnavi e Kuechler (2004).

- **Entendimento do problema:** a investigação é iniciada quando o pesquisador toma conhecimento de um problema ou da oportunidade de pesquisa.

- Sugestões: etapa criativa em que diferentes pesquisadores podem chegar a diferentes modelos experimentais. Durante a etapa de sugestão, elabora-se uma proposta de solução para o problema compreendido na etapa anterior. Estas sugestões são delineadas a partir da obtenção de conhecimento teórico e prático do problema objeto de estudo da pesquisa.

- **Desenvolvimento:** etapa onde o pesquisador construirá um ou mais artefatos. As técnicas utilizadas variam, dependendo do produto a ser construído.

- Avaliação: antes e durante o desenvolvimento do artefato, o pesquisador formula hipóteses sobre como será o seu comportamento. Quando desenvolvido, ele deve ser avaliado em função dos critérios definidos na proposta de trabalho, e mesmo desvios de expectativas devem ser relatados pelo pesquisador. Na DSR, raramente

hipóteses preliminares são desconsideradas. Entretanto, os desvios no desempenho esperado podem levar os pesquisadores a redefinir e buscar novas sugestões.

- Conclusões: são consolidados e registrados os resultados da pesquisa. Entretanto, não são apenas os resultados consolidados que devem ser registrados nesta etapa, mas também o conhecimento obtido é frequentemente classificado como proveitoso.

Visto que o *Design Science Research* é o método mais adequado à presente pesquisa, e compreendidas suas etapas propostas por Vaishnavi e Kuechler (2004), tem-se, a seguir, portanto, a aplicação do método estrutura o procedimento experimental da pesquisa em cinco fases, a saber: (i) problema e importância; (ii) proposta de solução; (iii) constructo: desenvolvimento e implementação; (iv) avaliação; e (v) conclusões.

Para a condução das fases do DSR nesta pesquisa, seguiu-se, os seguintes delineamentos de trabalho:

3.1 Problema e importância

Conforme detalha o capítulo 1 – Introdução, o problema e a importância desta pesquisa nascem da conscientização de que, embora a indústria AEC disponha de diversas tecnologias contemporâneas aplicadas a projetos, estas ainda são, muitas vezes, dissociadas e não preditivas, carecendo de instrumentos que conduzam o setor aos ganhos de produtividade e escala da era da inteligência artificial aplicada.

3.2 Proposta de solução

Este estudo propõe desenvolver um constructo, de maneira a contribuir para a inserção de algoritmos de IA no processo de elaboração dos projetos de edificações. Para a solução proposta nesta pesquisa foram aplicadas tecnologias disponíveis como o *design* generativo, a verificação automatizada de projetos paramétricos, o BIM e os algoritmos evolucionários em um único ambiente.

Como forma de obtenção de conhecimento teórico e prático, na revisão bibliográfica e na pesquisa as ferramentas disponíveis relativas às tecnologias os conceitos pertinentes foram ampliados e/ou revisados de maneira aplicável à problemática.

Durante a revisão bibliográfica deste trabalho, buscou-se por uma terminologia que pudesse retornar mais objetivamente a integração do universo BIM em face aos algoritmos evolucionários. Foi buscada a expressão “BIM Evolucionário” (ou “*Evolutionary* BIM”). Entretanto, até a data deste projeto de pesquisa[1], a terminologia não foi encontrada na busca de bases no Portal de Periódicos CAPES/MEC. Nem mesmo em busca livre pelo mecanismo de busca Google encontrou-se um resultado para a expressão exata.

3.2.1 BIM Evolucionário (E-BIM)

Observou-se, portanto, a oportunidade de proposição e adoção do termo “BIM Evolucionário” (E-BIM), como denominação deste constructo. A terminologia, ***a priori***, se demonstra pertinente uma vez que objetiva unificar os campos da computação evolucionária e dos projetos paramétricos, em especial os projetos BIM.

Para tanto, propõe-se “BIM Evolucionário” (E-BIM) como método aplicado a projetos de edificações, no qual modelos virtuais paramétricos são gerados, avaliados e otimizados pela integração de algoritmos generativos e evolucionários. Explora a aplicabilidade dos algoritmos de inteligência artificial no fluxo de desenvolvimento e automação destes projetos.

Em simetria à proposição deste conceito, a fim de se alcançar os objetivos geral e específicos deste estudo, o constructo E-BIM foi desenvolvido e implementado conforme detalha a seção a seguir.

[1] Projeto de Dissertação escrito até 07 de novembro de 2019. Banca de avaliação do Projeto de Dissertação ocorrida em 27 de novembro de 2019.

3.3 Constructo E-BIM

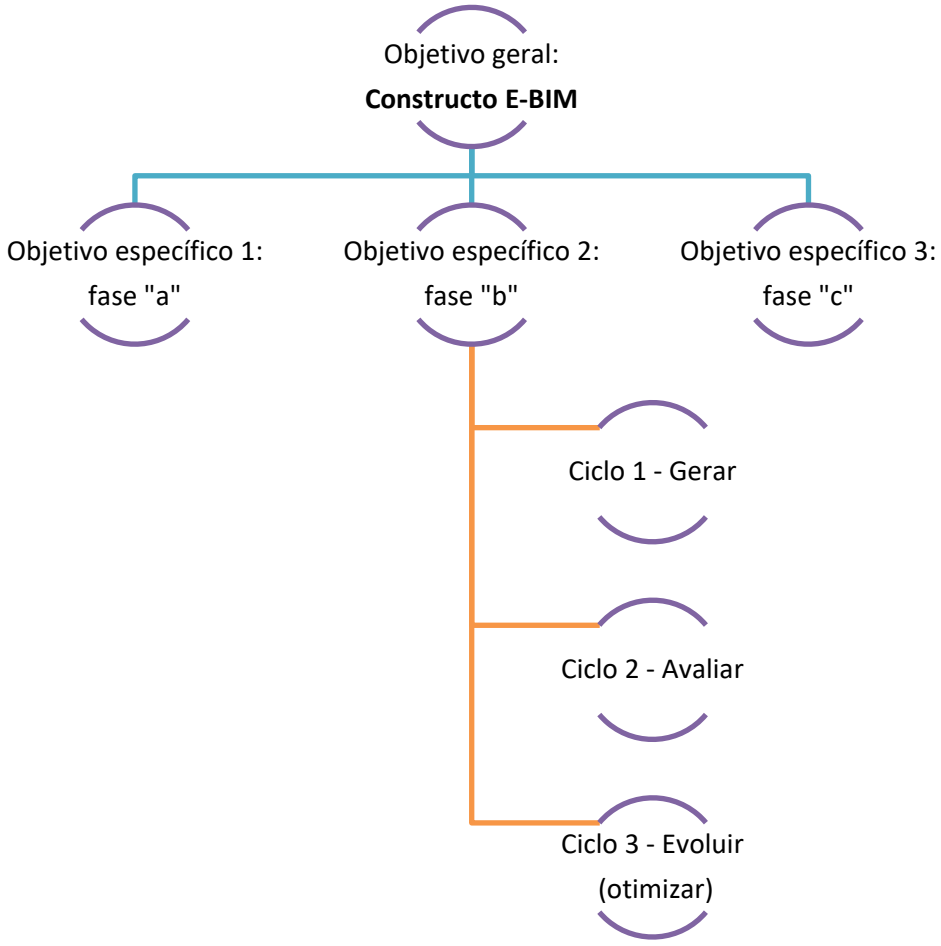
Para o cumprimento dos objetivos geral e específicos da pesquisa, o desenvolvimento e implementação do constructo foram planejados em três fases, conforme a Figura 16:

- **Fase A**: desenvolver, implementar e avaliar um algoritmo genético inserido no fluxo de projetos BIM, como instrumento evolucionário;

- **Fase B:** gerar, verificar e evoluir (otimizar) um objeto arquitetônico;

- **Fase C:** verificar a asserção do termo BIM Evolucionário (E-BIM) proposto ao contexto dos algoritmos de IA aplicados a projetos de edificações.

Figura 16 - Mapa do desenvolvimento e implementação do constructo E-BIM



Fonte: A autora (2020).

A “Fase A” constitui-se do desenvolvimento de um algoritmo evolucionário de otimização, através de um *script* Python, a fim de ser implementado e avaliado em um protótipo.

A “Fase B” contempla o modelo de implementação e avaliação do algoritmo da fase “a” e é construída em três ciclos, conforme padrão de um algoritmo evolucionário genético, sendo: (1) gerar; (2) avaliar; e (3) evoluir (otimizar).

Para o ciclo “Gerar”, ocorre a criação da população inicial de soluções de arquitetura, através de um algoritmo generativo, nos aplicativos Rhino-Grasshopper; uma amostragem desta população formará generativamente modelos paramétricos BIM do tipo IFC. No ciclo “Avaliar” são estabelecidos indicadores de medição de desempenho para a classificação dos indivíduos da amostra, que sejam mais aptos ao ciclo seguinte; no *software Solibri Model Checker* (SMC) foi desenvolvido e implementado um conjunto de regras de checagem automática (*ruleset*) desses indicadores e então os resultados exportados a partir de relatórios personalizados no SMC. No ciclo “Evoluir” é então executado o algoritmo evolucionário (fase “a”), novamente em ambiente Rhino-Grasshopper; o objetivo do algoritmo evolucionário é processar os indivíduos mais aptos da classificação, e interferir na geração de uma nova população, de maneira a otimizar a qualidade do processo generativo, de maneira que novos indivíduos contenham características superiores, ou computacionalmente evoluídas, em relação à geração inicial do *design* generativo, conforme os indicadores estabelecidos.

A “Fase C”, que diz que respeito à avaliação do constructo, é abordada na seção seguinte deste capítulo.

O desenvolvimento e a implementação do Constructo E-BIM estão detalhados no Capítulo 4.

3.4 Avaliação

Para Lukka (2003), inovar uma ideia de solução requer desenvolver uma construção de resolução de problemas e também trabalhar pela contribuição teórica. O autor adverte que este pode ser um processo iterativo trabalhoso e demorado, incluindo o desenvolvimento de ideias de protótipos, seus esforços de implementação em pequena escala e uma revisão na base do conhecimento. E aborda que, ainda que o teste falhe, haverá espaço para análises teóricas, pois provavelmente vale a pena discutir se as causas do insucesso poderiam ser evitadas em outros experimentos.

> A obtenção de conclusões teóricas interessantes não depende necessariamente se a nova construção projetada funcionou ou não: em ambos os casos, existe a possibilidade de contribuição teórica. Por exemplo, o que deveria ter funcionado com base na teoria anterior não funcionou - então, há algo errado com nossa teoria existente? No entanto, deve-se notar que uma pré-condição importante para tirar quaisquer conclusões significativas de uma pesquisa construtiva é que o trabalho empírico deve ser considerado bem administrado como um processo (LUKKA, 2003, p. 90).

Assim, em se tratando de uma pesquisa DSR, a avaliação do constructo E-BIM será suportada pela implantação de um protótipo. A prova de conceito terá como objeto uma arquitetura escolar, conforme exposição de motivações justificada na seção 4.1.1 deste trabalho.

Portanto, como estratégia de delineamento, a avaliação da contribuição prática do modelo deverá responder, ao menos, às questões apontadas na seção 1.3 deste trabalho.

Como anteriormente demonstrado por Lukka (2003), do ponto de vista acadêmico, a pesquisa DSR deve ser capaz de esclarecer também sua contribuição teórica, havendo, nesse caso, dois tipos principais de contribuições potenciais: I) se o novo constructo funcionou, poderá fornecer um incremento natural à literatura anterior, considerando-se as novas relações meio-fim construídas; II) além da tentativa de

projetar e testar novos instrumentos, o refinamento da teoria é, provavelmente, o resultado mais esperado de um projeto de pesquisa construtivo, pois fornece razão para modificar crenças anteriores a respeito das relações estruturais ou dos processos considerados antes de o estudo ter sido conduzido.

Isso posto, em continuidade à estratégia de delineamento, a avaliação da contribuição teórica deverá discorrer sobre a “Questão 4” do item 1.3 deste trabalho, que se refere à “Fase C” do constructo.

As questões então demarcadas, bem como suas conjunturas, são discutidas no capítulo 5 e sintetizadas no capítulo 6.

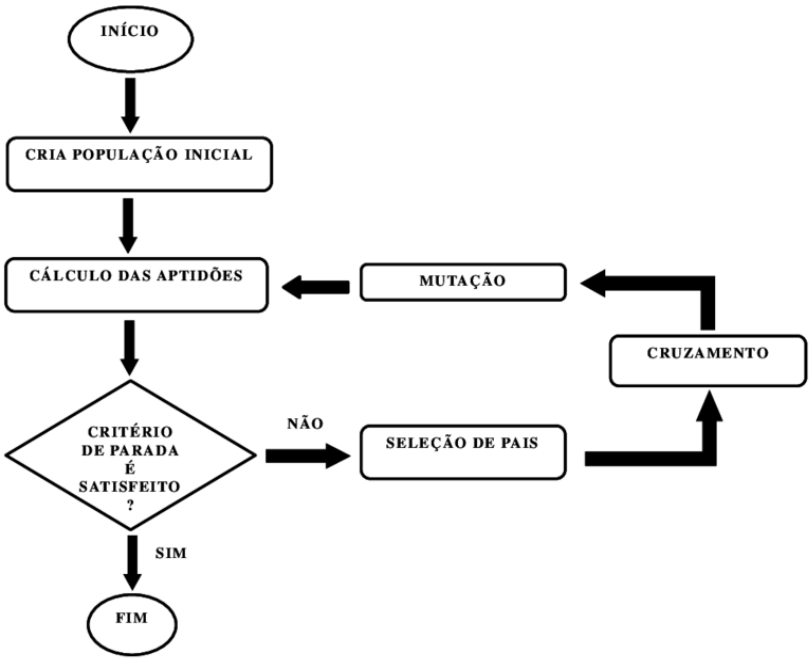
4 CONSTRUCTO: DESENVOLVIMENTO E IMPLEMENTAÇÃO

A presente pesquisa e dissertação de mestrado procurou desenvolver uma aplicação dos princípios evolutivos dos algoritmos de IA às tecnologias de projetos generativos e paramétricos BIM, tendo como objeto de implementação e avaliação um objeto arquitetônico de função escolar.

Na seção 2.4, viu-se que a Computação Evolucionária (CE) é um ramo de pesquisa da IA que propõe a solução de problemas inspirada na seleção natural, compreendendo um conjunto de técnicas de busca de otimização inspiradas na evolução natural das espécies. Diversas técnicas são utilizadas na CE, dentre elas os Algoritmos Genéticos (AG), que possuem ampla aplicação em muitas áreas científicas, entre elas a inteligência artificial e *machine learning.*

Na seção 2.4.1 foi compreendido o princípio dos Algoritmos Genéticos, que trata de gerar uma população de indivíduos que irão reproduzir e competir pela sobrevivência. Os mais aptos sobrevivem e transferem suas características às novas gerações, promovendo a evolução (aptidão) da espécie.

Figura 17 - Padrão de um Algoritmo Genético



Fonte: adaptado de Davis (1991).

Esse processo evolucionário pode ser sintetizado em três ciclos consecutivos (NAGY, 2017b):

1) Gerar: é criada uma população inicial de diferentes soluções do projeto objeto do estudo;

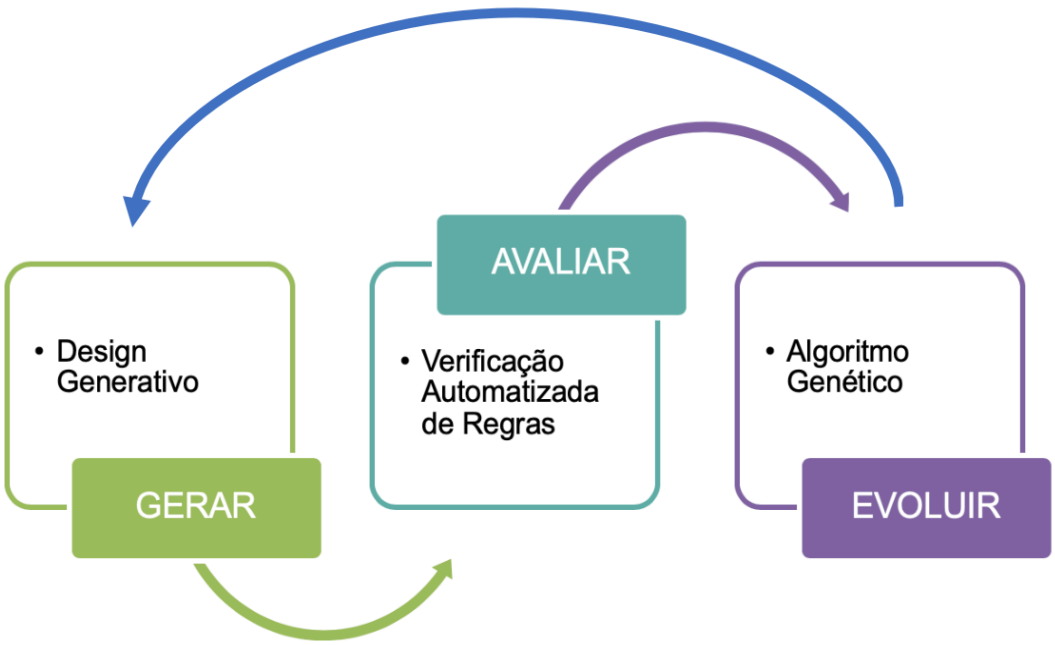
2) Avaliar: o grau de adequação das soluções individuais é calculado;

3) Evoluir: as soluções com melhor desempenho são levadas à uma operação genética (mutação, cruzamento) para gerar uma nova população otimizada.

A partir deste princípio, foram estruturados os processos de desenvolvimento das fases A e B do constructo (delineadas na seção 3.3), conforme representado na Figura 18.

Os desenvolvimentos dos ciclos “Gerar”, “Avaliar” e “Evoluir” estão descritos nas seções 4.1, 4.2 e 4.3 deste trabalho, respectivamente.

Figura 18 - Processo de desenvolvimento do constructo E-BIM



Fonte: A autora (2020).

4.1 Ciclo 1 - Gerar: *design* generativo

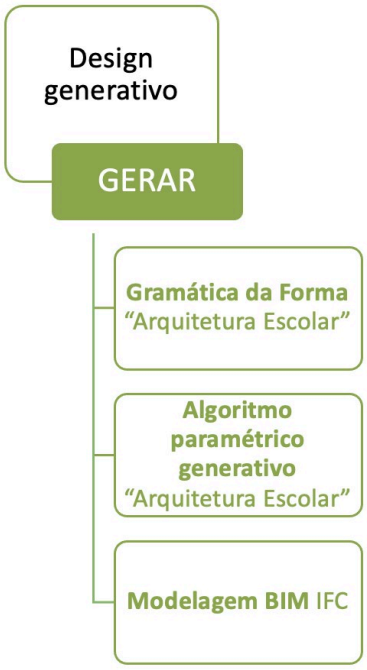
Baseado em algoritmos, o *software* Grasshopper tem por objetivo a geração de modelos paramétricos para arquitetura, *design* e áreas correlatas, como descrito na seção 2.1.2. Com uma linguagem de programação visual, o *software* é capaz de implementar algoritmos generativos e fornece funções de entrada e saída de dados para outros tipos de *softwares*, incluindo o sistema BIM (RMA, 2016).

Segundo Fischer e Herr (2001), um sistema generativo possibilita gerar conjuntos ou populações de um projeto, a partir da combinação exaustiva de seus parâmetros. A partir da geração de um grande número de alternativas para a solução de um programa de projeto, a possibilidade de obtenção de bons projetos é maximizada.

Segundo Stiny (1980), a Gramática da Forma é um sistema de geração de formas baseado em algoritmos parametrizados (computacionais ou não), em um vocabulário definido de formas geométricas e em um conjunto de regras de transformações euclidianas (translação, divisão, rotação, reflexão e escala), operadas nos locais definidos por marcadores. Assim, como descreve a seção 2.1.3, a Gramática da Forma é um método generativo de soluções formais.

Neste sentido, para o ciclo “Gerar” do desenvolvimento do constructo, foi elaborada uma gramática da forma para o objeto da prova de conceito (seção 4.1.1), e, a partir deste formalismo, foi desenvolvido um algoritmo paramétrico no Grasshopper (seção 4.1.2), de maneira a gerar a população inicial do algoritmo evolucionário. Desta população inicial retirou-se uma amostragem de indivíduos que, também de forma generativa, formaram modelos paramétricos BIM – em arquivos IFC, para o ciclo posterior “Avaliar” (seção 4.2). A Figura 19 representa essas subetapas.

Figura 19 - Subetapas do ciclo GERAR do constructo E-BIM



Fonte: A autora (2020).

4.1.1 Gramática da Forma "Arquitetura Escolar"

O objeto de prova de conceito do constructo E-BIM foi definido como um objeto arquitetônico de tipologia escolar, aqui denominado de "Arquitetura Escolar".

Não obstante, muitas vezes, tratada como conceito de conhecimento comum, a expressão "objeto arquitetônico" merece aqui uma breve revalidação: arquitetura como objeto para atender às necessidades humanas e abrigar as nossas atividades (NETTO, 2006).

Como mencionado no capítulo 1 deste trabalho, King (2017) quantifica a demanda por infraestruturas em escalas globais e evidencia urgência por soluções escaláveis na Construção Civil.

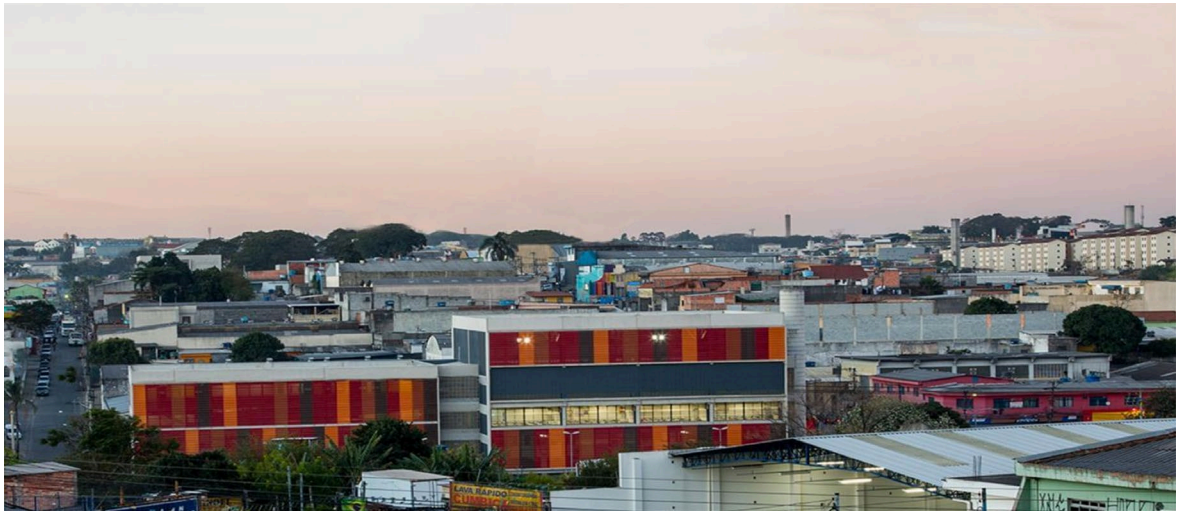
Com efeito, como prova de conceito deste constructo, poder-se-ia ter elegido um objeto arquitetônico de diversas tipologias e, portanto, optou-se por determinar uma tipologia escolar.

A orientação a esta tipologia deveu-se a dois fatores: 1) experimentalmente, como elemento de pesquisa, a tipologia escolar oferece mais complexidade em relação a um programa arquitetônico habitacional, por exemplo, o que permitiria testar maior alcance e aplicabilidades do constructo; 2) a questão social construtiva se relaciona amplamente com outras tipologias arquitetônicas e de infraestrutura que emergem da mesma problemática.

É importante lembrar aqui que, para as camadas mais carentes da população, que vivem nas periferias das cidades, a escola é praticamente o único equipamento público. Estas edificações se destacam por sua escala e arquitetura diferenciada em contraste à vastidão das autoconstruções e conjuntos habitacionais ainda massificados:

> [...] um prédio bem construído, funcional, agradável ao convívio e de melhor resultado plástico e espacial, poderá obter o reconhecimento dessa população, isto é, despertar o sentido de pertinência e identidade desses moradores com seu equipamento público, contribuindo para sua preservação. Irá permitir, também, que as crianças utilizem esse espaço e usufruam dessa referência de edifício, diferente daquele que habitam. Num sentido mais amplo, ao construir o prédio destinado à educação, considerando também questões sociais e ambientais, produz-se um ganho social e contribui-se para a formação da cidadania (FDE, 2006a, p.47).

Figura 20 - Escola Estadual Nova Cumbica (FDE)



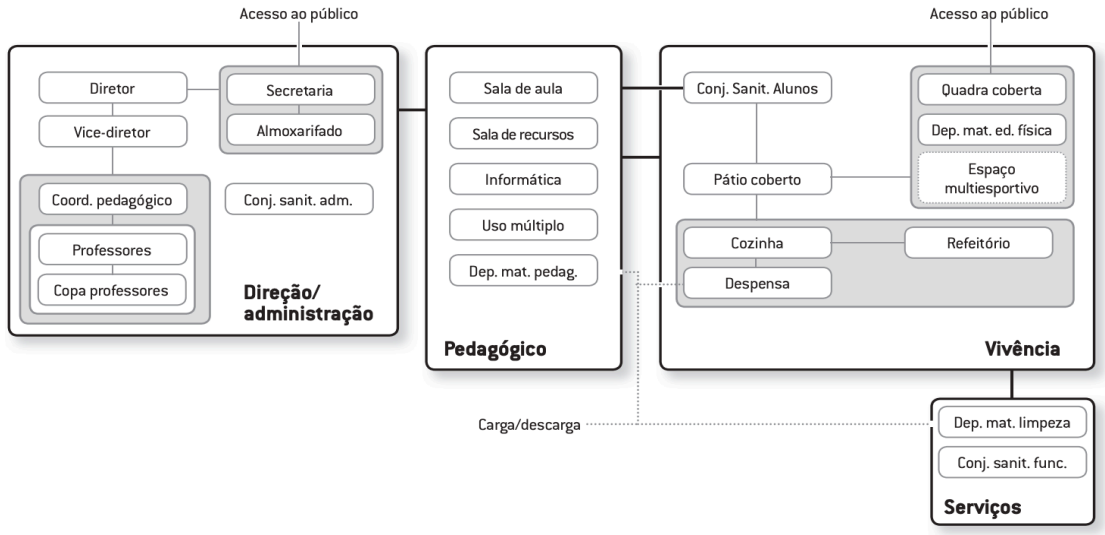
Fonte: FDE (2006a).

Nascimento (2012) coloca que o déficit educacional brasileiro levou à massificação do ensino e, durante boa parte do século XX, conduziu à adoção de soluções multiplicáveis, como os projetos-padrão e, posteriormente, a padronização de ambientes escolares. Segundo o autor, este procedimento contribuiu para a redução do potencial de inovação dos projetos, limitando a criação de novos espaços.

A Fundação para o Desenvolvimento da Educação (FDE, 2006) apresenta uma coletânea significativa de casos da arquitetura escolar pública paulista, e defende que é possível obter diversidade e qualidade plástica e espacial a partir de modulação e definições padronizados para o projeto.

A Gramática da Forma, elaborada para o sistema generativo do constructo E-BIM, referenciou-se em parâmetros[1] dos programas de projeto da FDE, por considerar que a Fundação tem alto reconhecimento de excelência em seus programas, sendo premiada em diversas áreas: programas e projetos, informática, mobiliário, vídeos, publicações e arquitetura. A Fundação conquistou pelo menos dezessete prêmios e menções em seus projetos arquitetônicos escolares (FDE, 2019a; FDE, 2019b).

Figura 21 - Fluxograma escolas FDE – anos iniciais do ensino fundamental



Fonte: FDE (2019).

[1] Programa de áreas conforme Anexo B e distribuição dos blocos conforme Figura 21.

Desta maneira, a Arquitetura Escolar, objeto da prova de conceito do constructo, contempla o programa distribuído em três blocos: pedagógico, administrativo e vivência.

a) Bloco 1 (B1) – Pedagógico: salas de aula e de usos diversos;

b) Bloco 2 (B2) – Administrativo: salas de direção e administração;

c) Bloco 3 (B3) – Vivência: pátio, vestiário (e sanitário), refeitório (e conjunto de serviços).

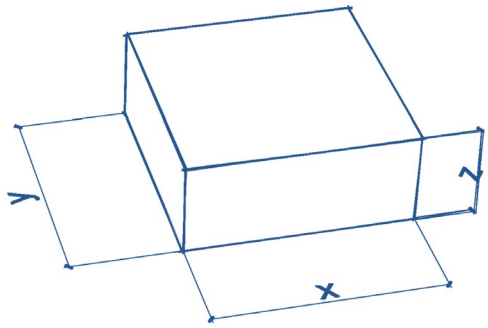
No vocabulário adotado para a Gramática da Forma, foram elaborados sete tipos de módulos (“s”, “c”, “h”, “e”, “p”, “v”, “r”), conforme Figura 22, que correspondem aos ambientes espaciais para atender o programa de arquitetura (Anexo B); combinados conforme as regras e marcadores descritos a seguir, serão as instâncias para o algoritmo generativo do ciclo “Gerar”.

Figura 22 - Módulos do vocabulário da Gramática da Forma

Fonte: A autora (2020).

Os módulos possuem funções e dimensões espaciais (*x*, *y*, *z*), conforme demonstram a Figura 23 e o Quadro 2:

Figura 23 - Dimensões espaciais dos módulos da Gramática da Forma



Fonte: A autora (2020).

Quadro 2 - Funções e dimensões dos módulos

MÓDULO	**AMBIENTE / FUNÇÃO**	**DIMENSÃO *x* (metros)**	**DIMENSÃO *y* (metros)**	**DIMENSÃO *z* (metros)**
s	sala	6,0	8,5	3,5
c	circulação horizontal	2,5	8,5	3,5
h	circulação vertical	8,5	8,5	3,5
e	quadra poliesportiva	15,0	46,0	7,0
p	pátio	8,5	8,5	3,5
v	vestiário	4,25	4,25	3,5
r	refeitório	17,0	8,5	3,5

Fonte: A autora (2020).

Para cumprir o programa de áreas do programa de necessidades, as quantidades e agrupamentos dos módulos seguiram a ordenação do Quadro 3:

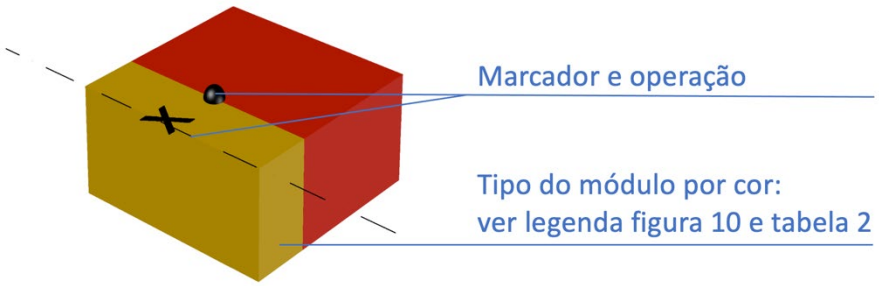
Quadro 3 - Quantidades e agrupamentos dos módulos

BLOCO	QUANTIDADE	MÓDULOS
B1	10	
	10	
B2	5	
	5	
B3	1	
	4	
	4	
	1	

Fonte: A autora (2020).

Para combinar e operar os módulos demonstrados nas legendas da Figura 22 e dos Quadros 2 e 3, foram criadas 12 regras que são classificadas entre obrigatórias, condicionais e aleatórias. E para que os algoritmos possam operar tais regras, localizam-se nos módulos os marcadores onde as operações ocorrem. A Figura 24 demonstra tipicamente esta ordenação:

Figura 24 - Representação típica de módulo e ordenação de regras



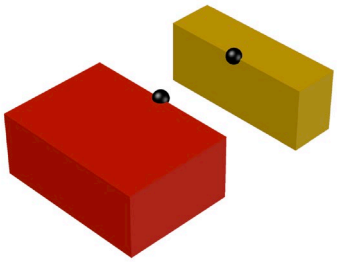
Fonte: A autora (2020).

A seguir, são descritas as 12 regras da Gramática da Forma:

i. **Regra 1**

Classe: obrigatória

Módulos e Marcadores:

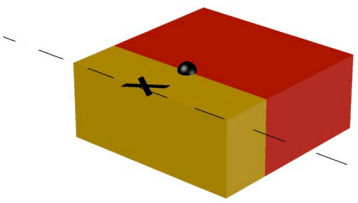


Operação: ● atrai e conecta ●

ii. **Regra 2**

Classe: aleatória

Módulos e Marcadores:

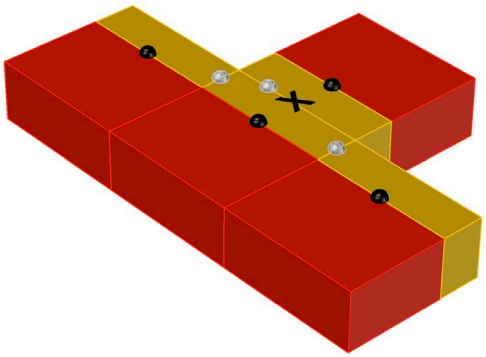


Operação: X espelha na aresta

iii. **Regra 3**

Classe: aleatória

Módulos e Marcadores:

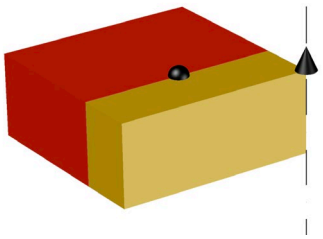


Operação: ● atrai e conecta ● , no eixo *x* ou no eixo *y*

iv. **Regra 4**

Classe: aleatória

Módulos e Marcadores:

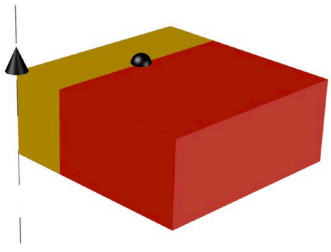


Operação: ▲ gira 90º na aresta do eixo *z*

v. **Regra 5**

Classe: aleatória

Módulos e Marcadores:

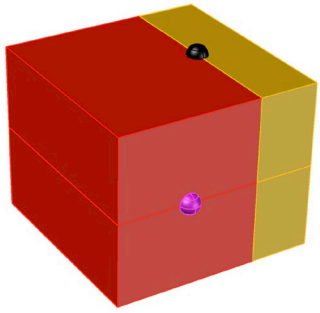


Operação: ▲ gira 270º na aresta do eixo *z*

vi. **Regra 6**

Classe: aleatória

Módulos e Marcadores:



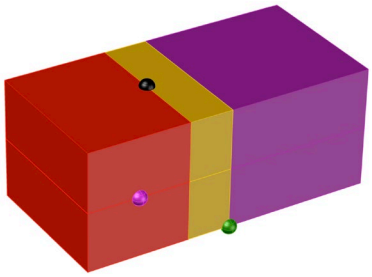
Operação: ● atrai e conecta ● , no eixo *z*

vii. **Regra 7**

Classe: condicional

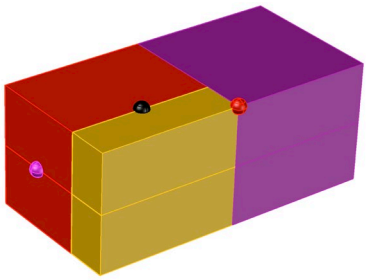
- Se ocorrer Regra 6, então haverá a Regra 7.1 ou 7.2 ou 7.3, onde:

- Regra 7.1
  Módulos e Marcadores:



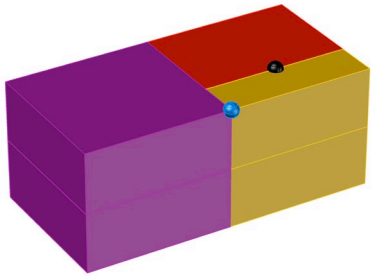
Operação: ● atrai e conecta ●

- Regra 7.2
  Módulos e Marcadores:



Operação: ● atrai e conecta ●

- Regra 7.3
  Módulos e Marcadores:

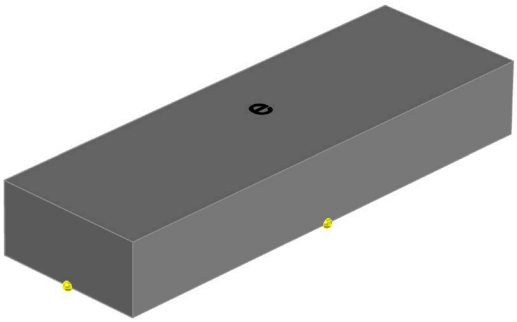


Operação: ● atrai e conecta ●

viii. **Regra 8**

Classe: obrigatória

Módulos e Marcadores:

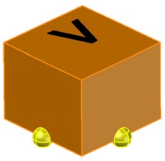


Operação: ● *e* atrai e conecta ● *v*

ix. **Regra 9**

Classe: obrigatória

Módulos e Marcadores:

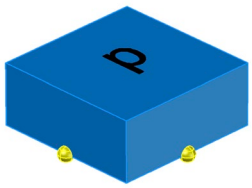


Operação: ● *v* atrai e conecta ● *v*
● *v* atrai e conecta ● *p*

x. **Regra 10**

Classe: obrigatória

Módulos e Marcadores:

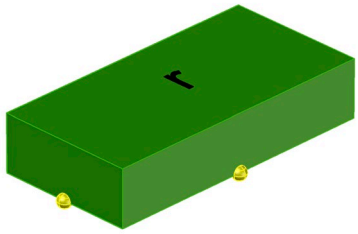


Operação: ○ *p* atrai e conecta ○ *p*

xi. **Regra 11**

Classe: obrigatória

Módulos e Marcadores:



Operação: ○ *r* atrai e conecta ○ *p*

xii. **Regra 12**

Classe: obrigatória, onde:

- Bloco 1 (B1) conecta Bloco 2 (B2) e Bloco 1 (B1) conecta Bloco 3 (B3)

ou,

- Bloco 1 (B1) conecta Bloco 2 (B2) e Bloco 2 (B2) conecta Bloco 3 (B3)

e se,

- coordenada *z* de B1 ≠ coordenada *z* de B2: operar também Regra 7
- coordenada *z* de B1 ≠ coordenada *z* de B3: operar também Regra 7
- coordenada *z* de B2 ≠ coordenada *z* de B3: operar também Regra 7

Os blocos descritos na Regra 12 dizem respeito aos agrupamentos do programa de necessidades, conforme Figura 21 e Quadro 3.

A partir das instâncias definidas nesta Gramática da Forma, foi implementado o algoritmo paramétrico generativo no Grasshopper.

4.1.2 Algoritmo paramétrico generativo "Arquitetura Escolar"

Em continuidade ao ciclo "Gerar" do constructo, após a estruturação do sistema generativo com a criação da Gramática da Forma (seção 4.1.1), foi implementado o algoritmo no ambiente computacional Rhinoceros-Grasshopper. Esta e todas as especificações de equipamentos e *softwares* utilizadas no experimento desta pesquisa estão descritas no Apêndice A. Os *softwares* e *hardwares* utilizados pela autora na pesquisa foram autopatrocinados, viabilizando a abrangência multidisciplinar e cronograma do trabalho.

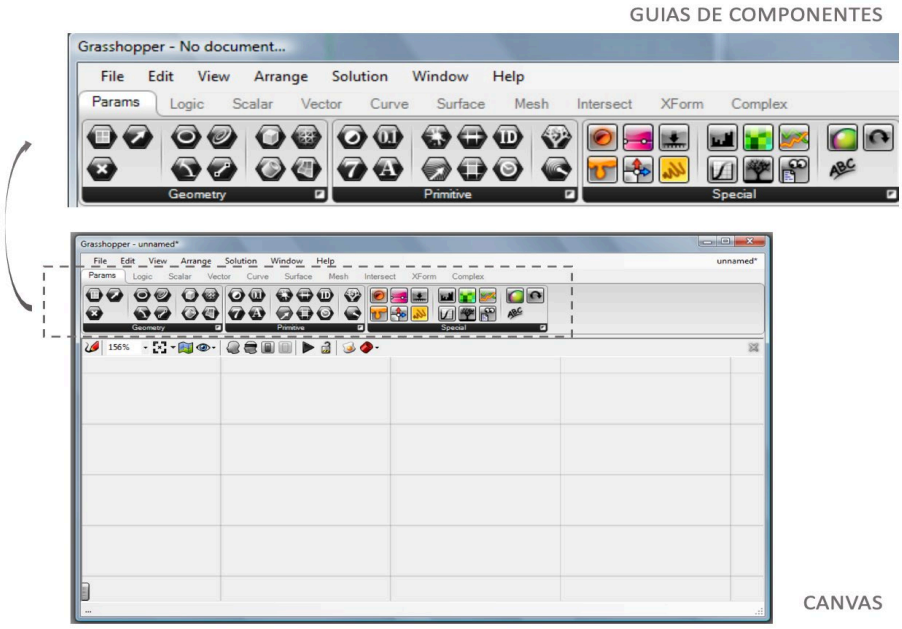
Conforme descrito na seção 2.1.2, com a integração dos dois *softwares* é possível aplicar computacionalmente as regras de geração das formas de forma dinâmica e interativa à visualização 3D.

Gramáticas paramétricas podem gerar uma população muito grande de combinações de formas viáveis, permitindo a exploração de alternativas mais complexas e até imprevisíveis. Métodos de *design* generativo são ferramentas precisas, flexíveis e controláveis que permitem alterações com mais facilidade.

A interface de programação visual do Grasshopper permite que o desenvolvimento do algoritmo seja visto como um fluxograma. Possui menus similares ao sistema Windows e duas partes importantes na interface: guias de componentes e um painel (canvas), conforme apresenta a Figura 25. As guias de componentes fornecem os elementos necessários para programação do algoritmo ou da geometria – sendo estes componentes nativos ou *plug-ins* instalados; o painel (canvas) é o local de trabalho onde são carregados os componentes e estruturado o algoritmo do *design*.

Um algoritmo pode ser compreendido como uma rotina de tarefas em ordem: recebe informações, processa dados e entrega resultado. A Figura 26 demonstra um algoritmo generativo simplificado.

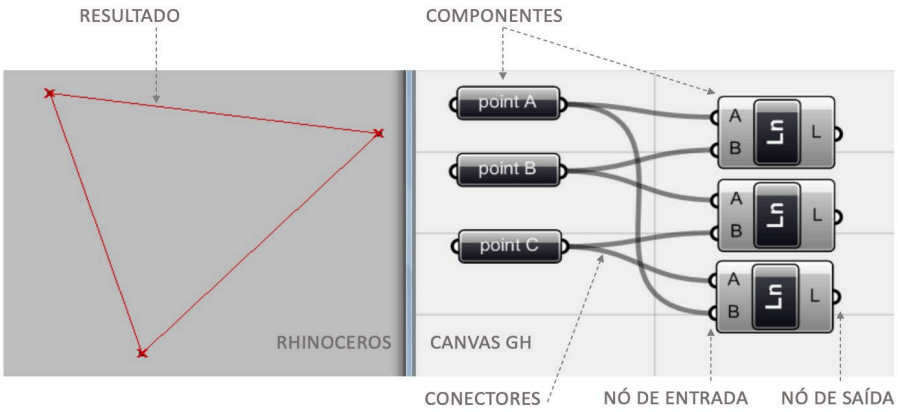
Figura 25 - Guias de componentes e painel (canvas) Grasshopper



Fonte: A autora (2020).

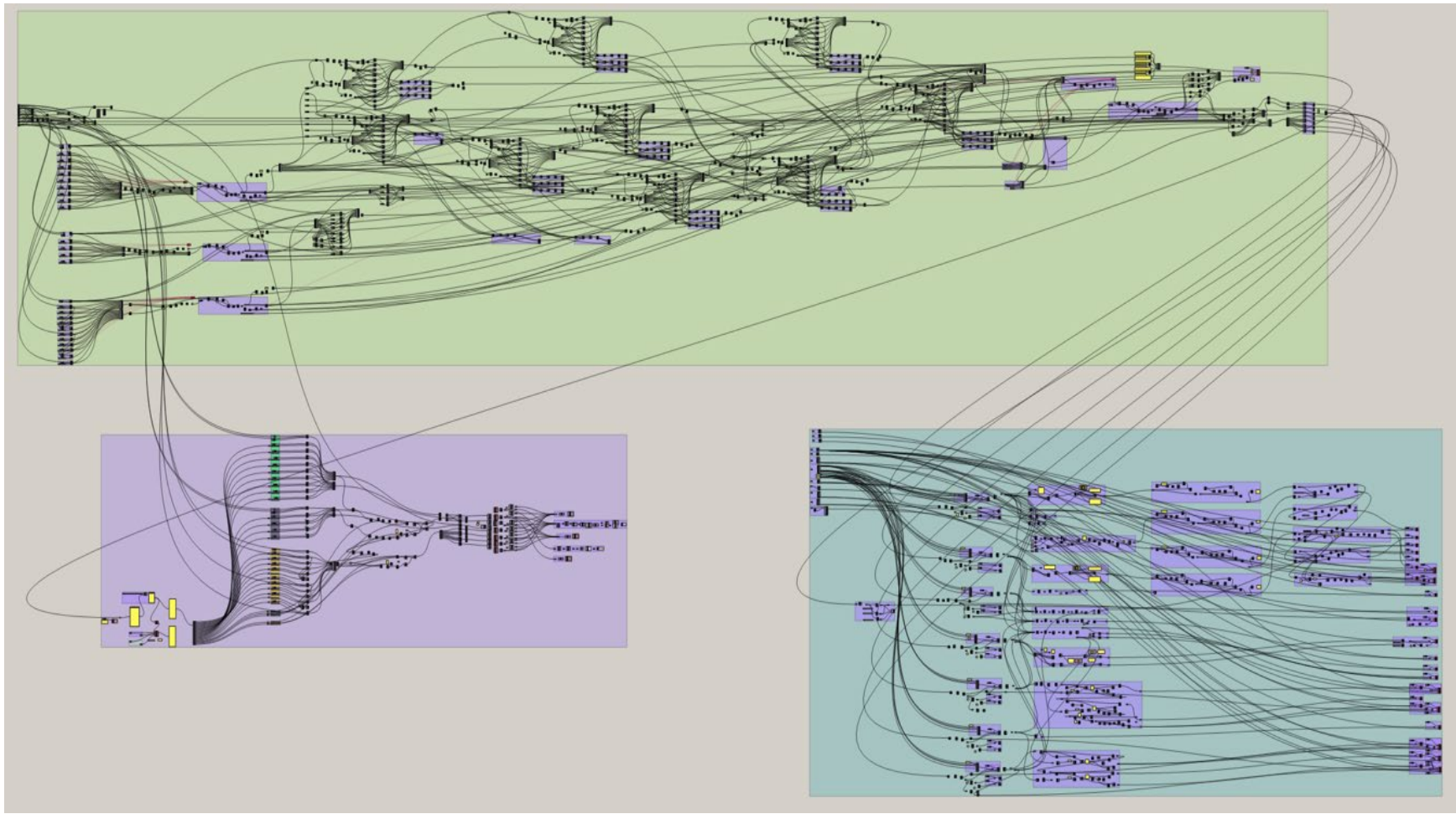
Para configurar um algoritmo de projeto, é necessário fornecer dados para diversos componentes, conectá-los na ordem da tarefa que se deseja executa e obter o resultado. Assim, um algoritmo generativo é composto de vários componentes com conectividade lógica (KHABAZI, 2012).

Figura 26 - Organização de um algoritmo generativo no Grasshopper



Fonte: A autora (2020).

Figura 27 - Algoritmo generativo completo e partes do desenvolvimento

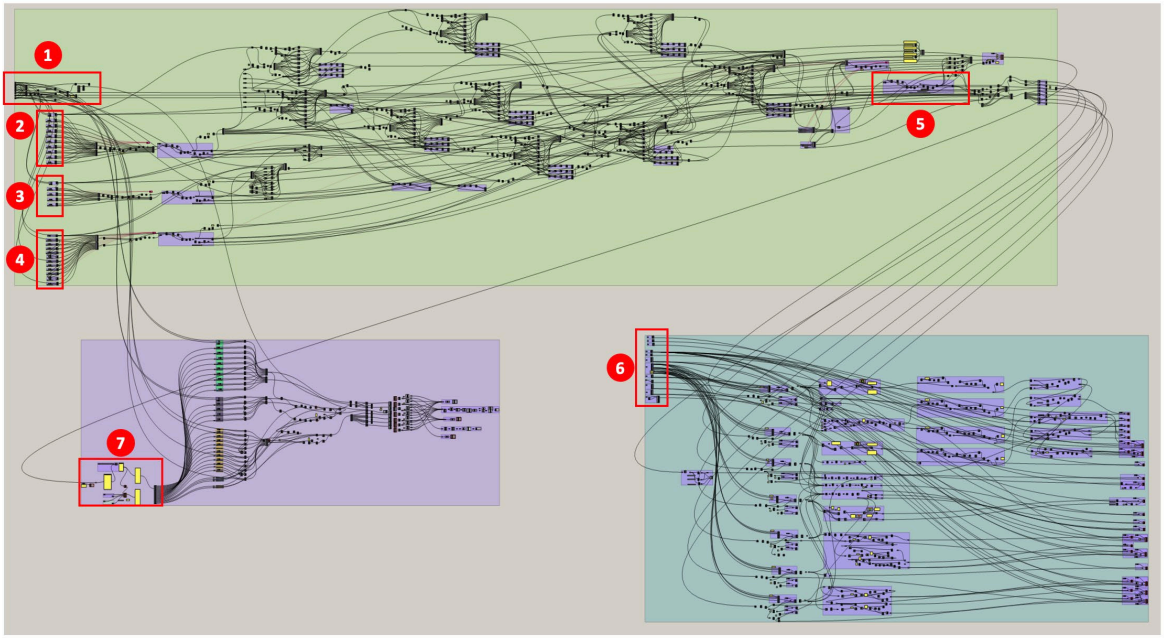


Fonte: A autora (2020).

O algoritmo foi desenvolvido em três partes: i) Gramática da Forma; ii) geração BIM IFC; iii) algoritmo evolucionário; esquematizadas na Figura 27 nas cores verde, azul e roxo, respectivamente.

Como indicado na Figura 27, o fluxograma completo do algoritmo é altamente complexo e extenso[1]. Por esta razão, as partes mais relevantes do desenvolvimento serão demonstradas em 7 destaques, conforme ordena a Figura 28 a seguir.

Figura 28 - Destaques do algoritmo generativo



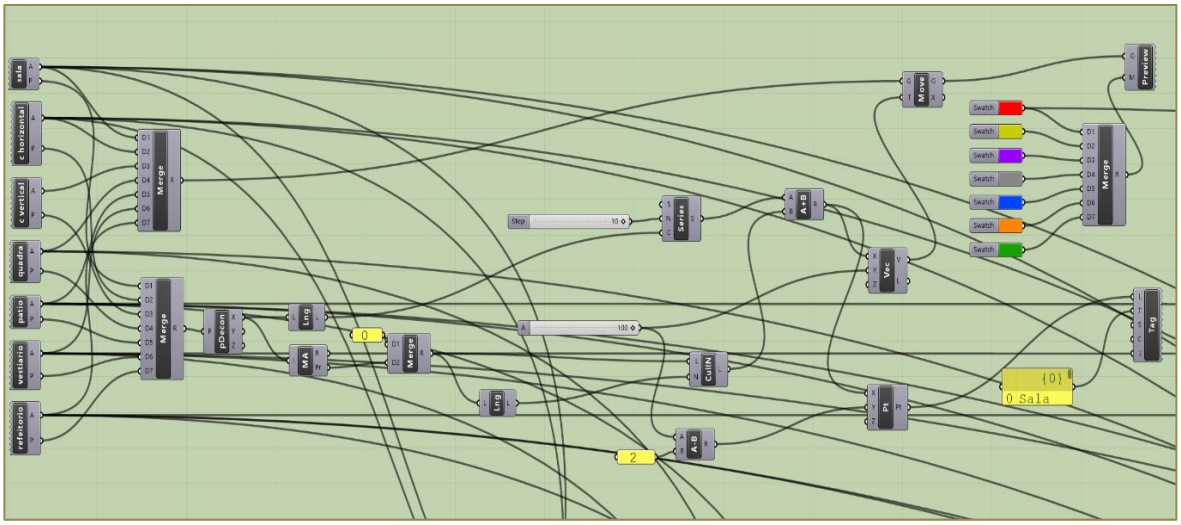
Fonte: A autora (2020).

- Destaque 1: formação geométrica dos ambientes (módulos)

Conforme ilustra a Figura 29, o algoritmo generativo é iniciado organizando-se os módulos e as regras que irão operá-los. Cada denominação de módulo (ver Figura 22/Quadro 3) é um agrupamento (*cluster*) de funções do algoritmo. Os grupos de funções aparecem como um único componente no espaço de trabalho do Grasshopper.

[1] A leitura impressa, mesmo em um formato estendido (A1, por exemplo), foi testada e não apresentou legibilidade satisfatória.

Figura 29 - Destaque 1 do algoritmo generativo



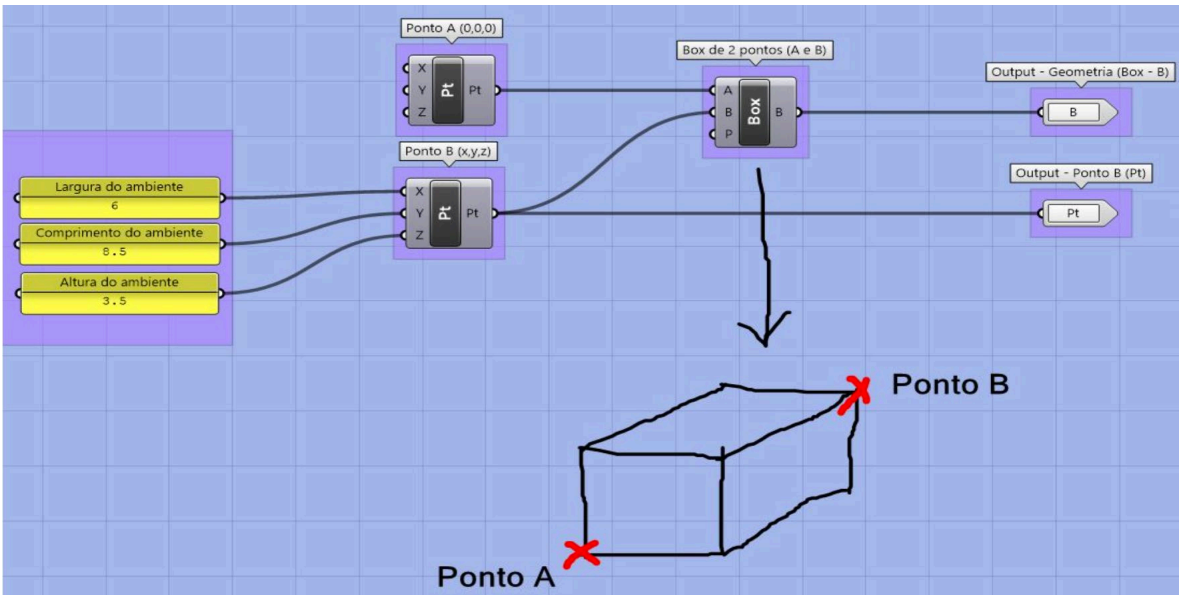
Fonte: A autora (2020).

Para permitir que o algoritmo funcione de forma otimizada, sem arquivo secundário do Rhinoceros, os ambientes (módulos) foram modelados diretamente no Grasshopper como “caixas”, através do componente “*Box* 2pt”, usando-se como referência 2 pontos localizados nas extremidades da caixa (o ponto 0,0,0 e o ponto com as coordenadas *x, y* e *z* correspondentes ao comprimento, largura e altura do ambiente, conforme Quadro 3).

A Figura 30 apresenta o interior do *cluster* “Sala”, com a formação geométrica do ambiente (módulo) definido na Gramática da Forma. O mesmo processo se repetiu para os *clusters* “Circulação Horizontal”, “Circulação Vertical”, “Quadra”, “Pátio”, “Vestiário” e “Refeitório”.

Para efeitos de visualização dos ambientes em 3D no Rhinoceros, um bloco de componentes foi criado, distribuindo os ambientes de forma sequencial. Os ambientes foram agrupados, componente “*Merge*”, e receberam uma cor, de maneira a corresponder visualmente à legenda da Figura 22.

Figura 30 - Interior do *cluster* “sala”



Fonte: A autora (2020).

Após configurações dos parâmetros iniciais (elementos da gramática), foram desenvolvidas as simulações das regras da Gramática da Forma. Neste momento, são organizados os agrupamentos das regras de acordo com a formação dos blocos B1, B2 e B3, como determina o Quadro 3.

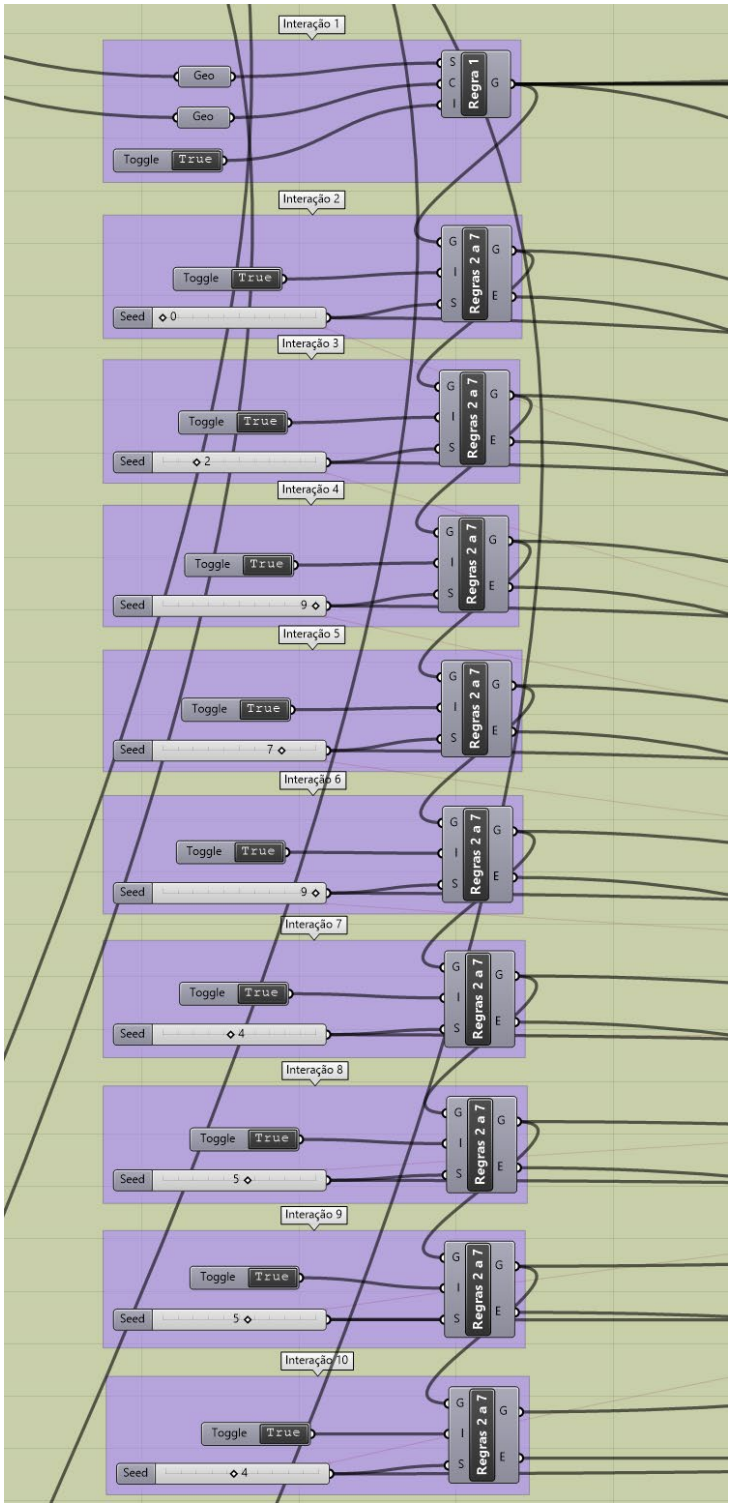
- Destaques 2 e 3: geração dos blocos B1 e B2

Os blocos B1 e B2 possuem parâmetros similares, exceto pelo número de ambientes, logo, as informações relacionadas a B1 também atendem a B2.

O algoritmo é estruturado em um sistema de interações onde cada interação é baseada em dois *inputs*: i) o resultado da interação anterior; e ii) um “*Seed*” (geração) randômico. As interações estão agrupadas em *clusters*, sendo a primeira interação diferente das demais, pois nela ocorre somente a execução da regra 1 (da Gramática da Forma), havendo a criação do módulo integrado “Sala + Circulação Horizontal”. Nas próximas interações acontece o processamento das regras 2 a 7, adicionando-se um novo módulo integrado ao módulo da interação anterior. A regra a ser aplicada na criação desse novo módulo é baseada em um valor randômico. Tais interações acontecem num total de 10 vezes para o grupo B1

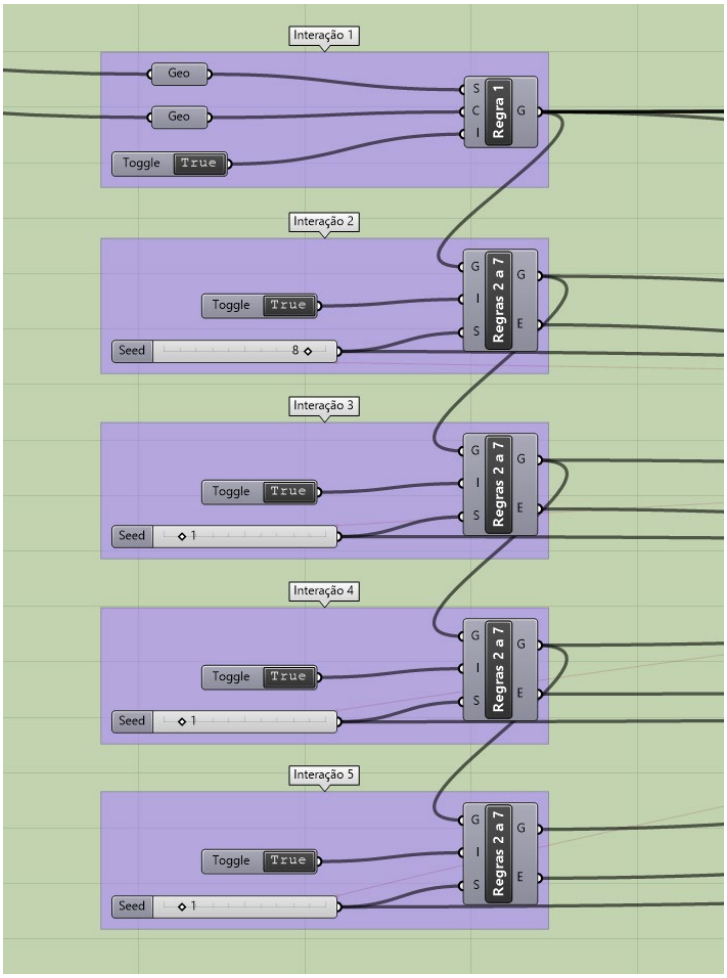
e de 5 vezes para o grupo B2, correspondendo à quantidade de módulos estipulada para estes blocos no Quadro 3. As Figuras 31 e 32 representam estas formações dos grupos B1 e B2, respectivamente:

Figura 31 - Destaque 2: interações B1



Fonte: A autora (2020).

Figura 32 - Destaque 3: interações B2



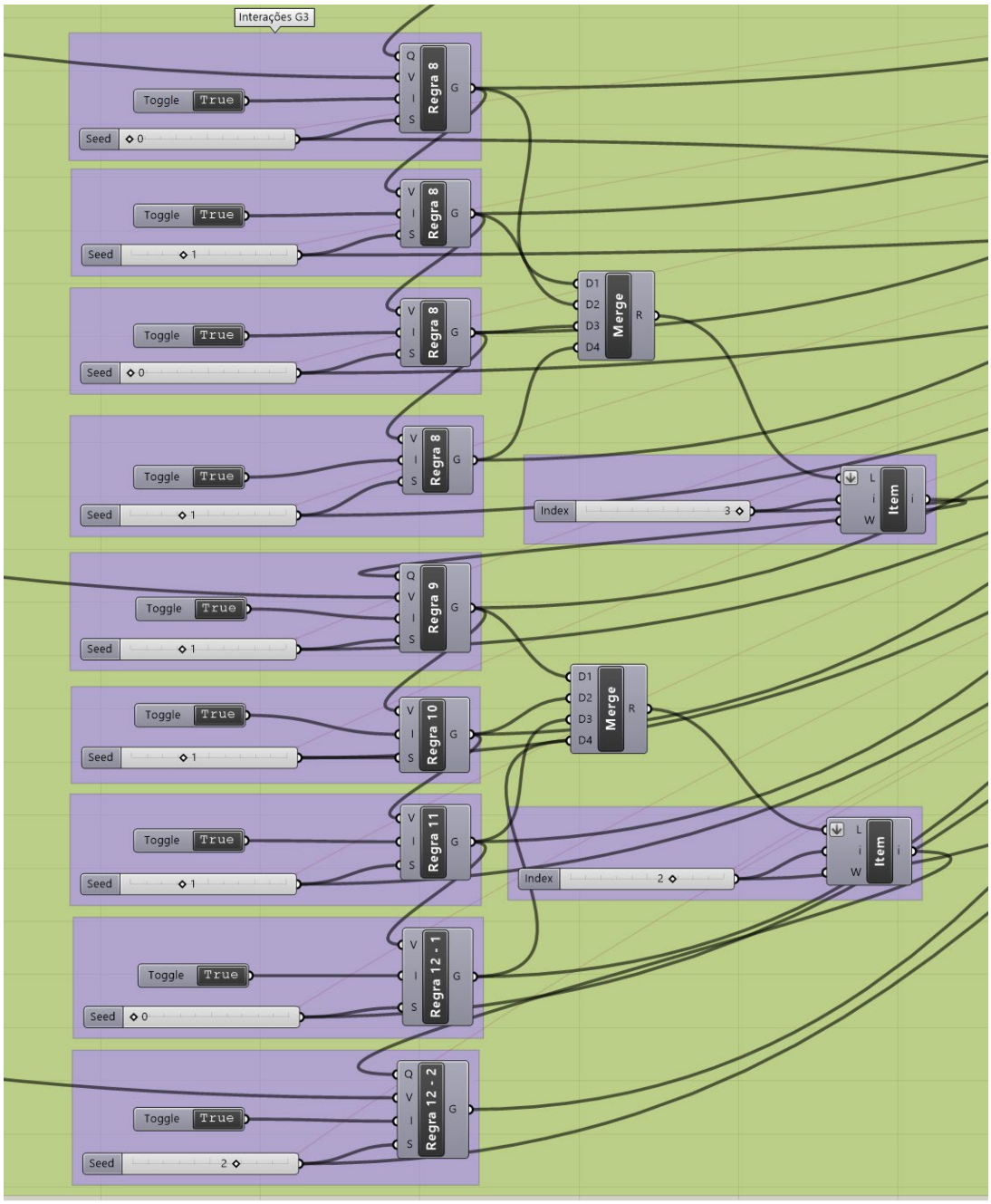
Fonte: A autora (2020).

- Destaque 4: geração do bloco B3

A geração do bloco B3 também é baseada em interações e regras, entretanto, a estruturação do código tem fundamentos diferentes a respeito de ordem de influência umas nas outras. As regras que compõem este grupo (regras 8 a 12) são hierarquicamente mais importantes do que as interações por serem todas obrigatórias (assim como a regra 1). Aqui as interações têm a função de inserir variáveis randômicas sobre a determinação das conexões entre os módulos.

Assim como nas interações, as regras ilustradas na Figura 33 também estão organizadas em *clusters*.

Figura 33 - Destaque 4: interações B3



Fonte: A autora (2020).

Durante o desenvolvimento do algoritmo foi observado que, mesmo ao seguir essas regras, os ambientes (módulos) poderiam se colidir (sobrepor), ocupando algum ponto espacial em comum, frustrando o processo generativo. Uma solução possível para evitar esse erro seria a criação de diversas condicionais em que, por exemplo, uma determinada regra seria invalidada no caso de ocorrer a sobreposição. Em outras palavras: criação de subregras de lógica que condicionassem o algoritmo.

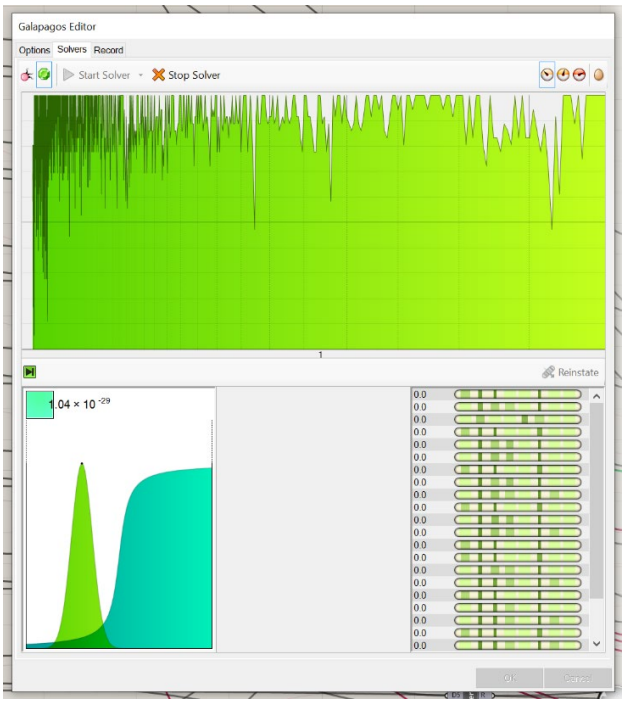
Entretanto, ao se impor condicionais, o código tornava-se cada vez mais complexo e lento, com grande ocorrência de falhas (*bugs*). A melhor estratégia, então, foi permitir que os parâmetros gerassem milhares de combinações – válidas ou não, integrando estas gerações a um mecanismo de filtragem de colisões. Assim, o algoritmo funcionou gerando o máximo de combinações possíveis e descartando as que não são desejáveis (*outputs* com colisões detectadas).

Desta forma, após as combinações geradas para os blocos B1, B2 e B3, foram realizadas a simulação e a filtragem dos resultados parciais do algoritmo. Estes processos foram executados com o recurso Galapagos (atualmente já nativo ao *plug-in* Grasshopper).

Ilustrado pela Figura 34, o Galapagos é um solucionador algorítmico baseado em critério similar ao critério evolutivo, em que uma população de soluções candidatas é mantida e novas soluções são geradas aleatoriamente por mutação ou recombinação de variantes na população existente. Ciclicamente, a população gerada é cortada pela aplicação de um critério de seleção (uma função de adequação – *fitness*), permitindo que apenas os melhores candidatos sobrevivam na próxima geração. Iterando ao longo de muitas gerações, a qualidade média das soluções aumenta gradualmente.

Neste caso, a função de adequação desejada (*fitness*) foi relacionada ao número de colisões. O recurso "*Solver*" do Galapagos buscou as soluções de composição volumétrica que retornavam esse valor de colisões igual a zero.

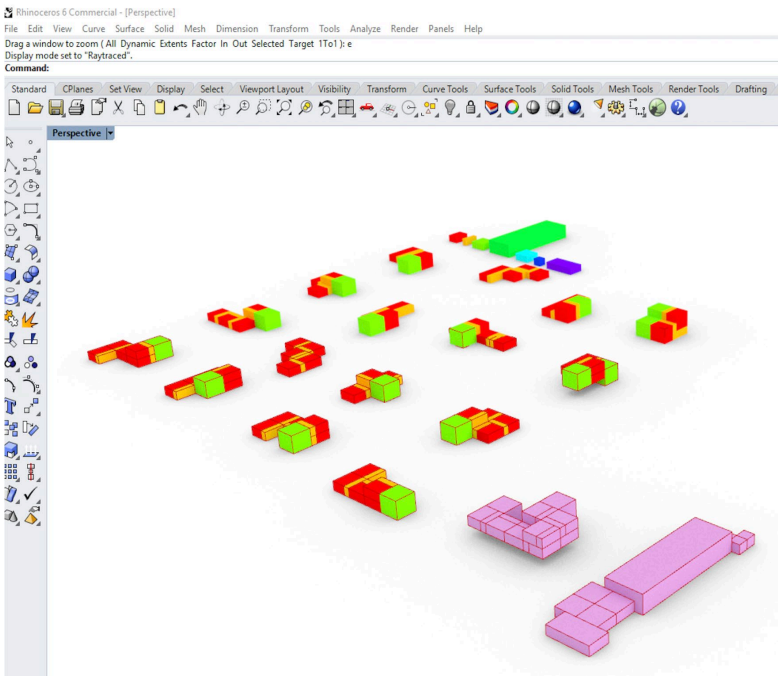
Figura 34 - Solucionador algorítmico Galapagos



Fonte: A autora (2020).

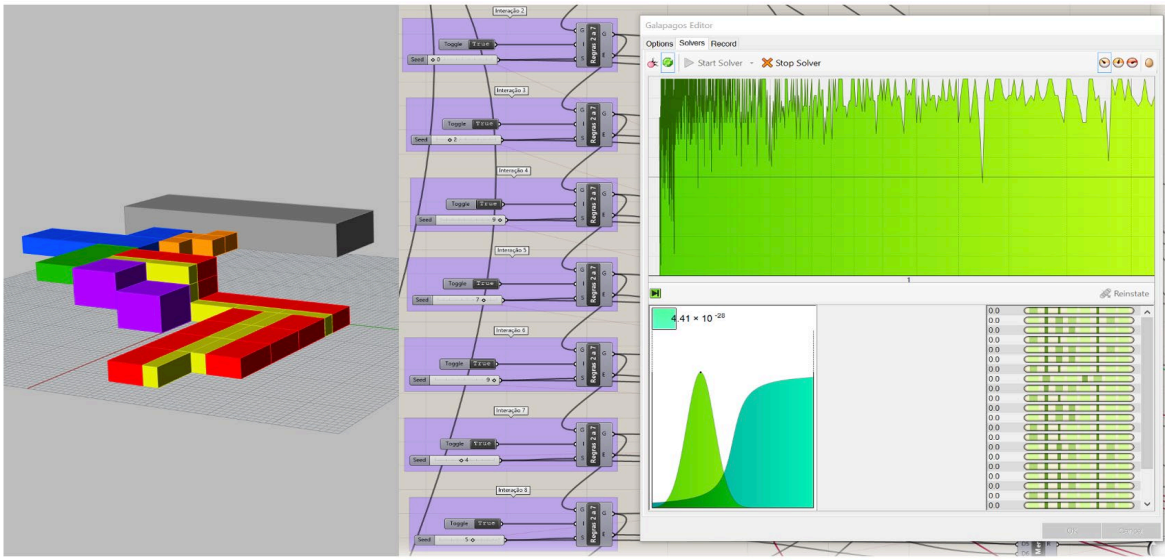
As Figuras 35 e 36 são capturas ilustrativas do processo de formação volumétrica dos indivíduos aptos, na integração Galapagos, Grasshopper e Rhinoceros.

Figura 35 - Geração de indivíduos aptos



Fonte: A autora (2020).

Figura 36 - Integração Galapagos, Grasshopper e Rhinoceros



Fonte: A autora (2020).

- Destaque 5: filtragem e armazenagem dos indivíduos aptos

Cada solução sem colisão encontrada pelo Galapagos é então denominada “indivíduo apto da população”. Após esta busca, foi incluído um processo de filtragem e armazenagem dos indivíduos aptos, com seus respectivos “códigos genéticos”[1].

Esse destaque 5 do algoritmo generativo, representado na Figura 37, trata de um conjunto de processos que encerra a parte (i) do algoritmo, que, por sua vez implementa a codificação da Gramática da Forma.

Como discorrido até aqui, a combinação das regras da Gramática da Forma com instrumentos de condicionamento de soluções válidas, em um processamento computacional atual, é capaz de gerar um número vasto de soluções formais para uma mesma proposição.

Neste projeto não foi estipulado um processo automático de parada de processamento do algoritmo. Nas simulações de testagem do algoritmo observou-

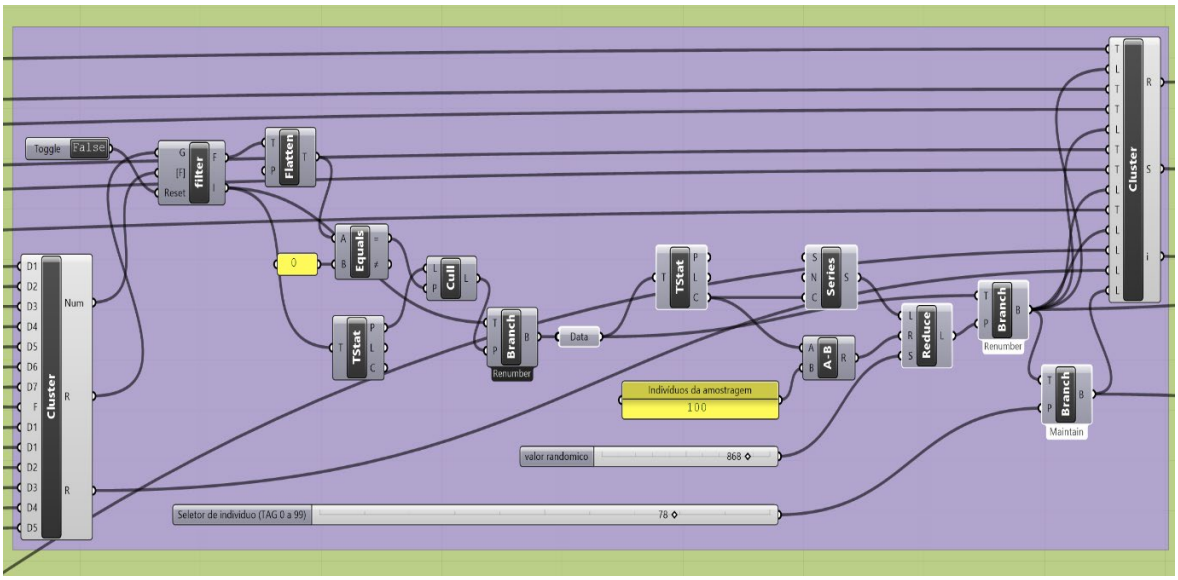
[1] Na seção 4.3 o conceito de “código genético” será retomado contextualmente.

se que é possível a coleta de uma quantidade mínima de soluções mesmo em um pequeno espaço de tempo.

Assim, “tempo” foi o critério de parada definido para coleta de resultados de implementação desta fase do algoritmo generativo. O experimento foi processado durante precisamente 12 horas, com o computador totalmente dedicado à tarefa. As especificações do equipamento utilizado estão descritas no Apêndice A. Nestas condições, portanto, foram coletadas soluções válidas criadas generativamente pelo algoritmo.

Número de soluções válidas coletadas em 12 horas de processamento: **8.574**. Desta população, baseado em processo de seleção aleatória randômica, foi retirada uma amostragem de 100 indivíduos que alimentaram as partes subsequentes do experimento.

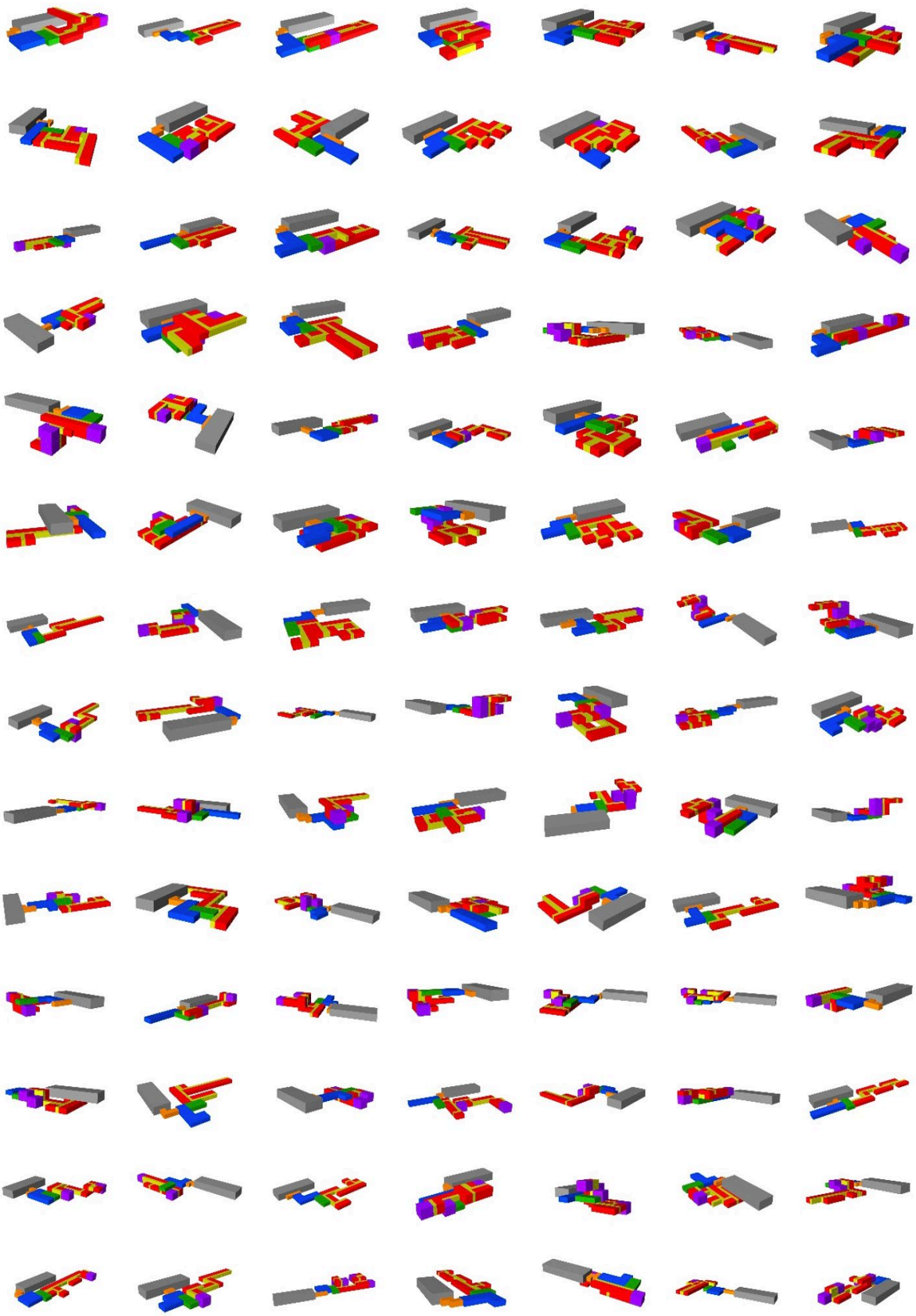
Figura 37 - Destaque 5: filtragem e coleta da amostragem



Fonte: A autora (2020).

A galeria da Figura 38 apresenta, esquematicamente, 98 (de 100) indivíduos da amostragem da população, que podem ser melhor examinados no Apêndice B. Cada indivíduo recebeu um nome denominado “TAG n”, onde n é sua numeração na população. Conforme padrão do Grasshopper, a numeração começa em zero. Logo, os indivíduos foram nomeados TAG0 a TAG99.

Figura 38 - Amostragem de indivíduos da população



Fonte: A autora (2020).

4.1.3 Modelagem BIM IFC

- Destaque 6: modelagem BIM IFC generativa

Como abordado na seção 2.1.1 deste trabalho, em termos de eficiência no fluxo do projeto, a introdução da tecnologia BIM permitiu a modelagem da realidade virtual da construção e a extração de todas as informações de projeto, desenhos e representações, a qualquer momento, além de apoiar todas as partes envolvidas nos processos de projeto e construção.

Entretanto, mesmo que o BIM seja definido como uma abordagem poderosa para o processo de projeto, ainda não é capaz de otimizar a questão de tarefas repetitivas, muitas vezes lentas, sobretudo em produção de geometrias mais complexas ou de grandes áreas.

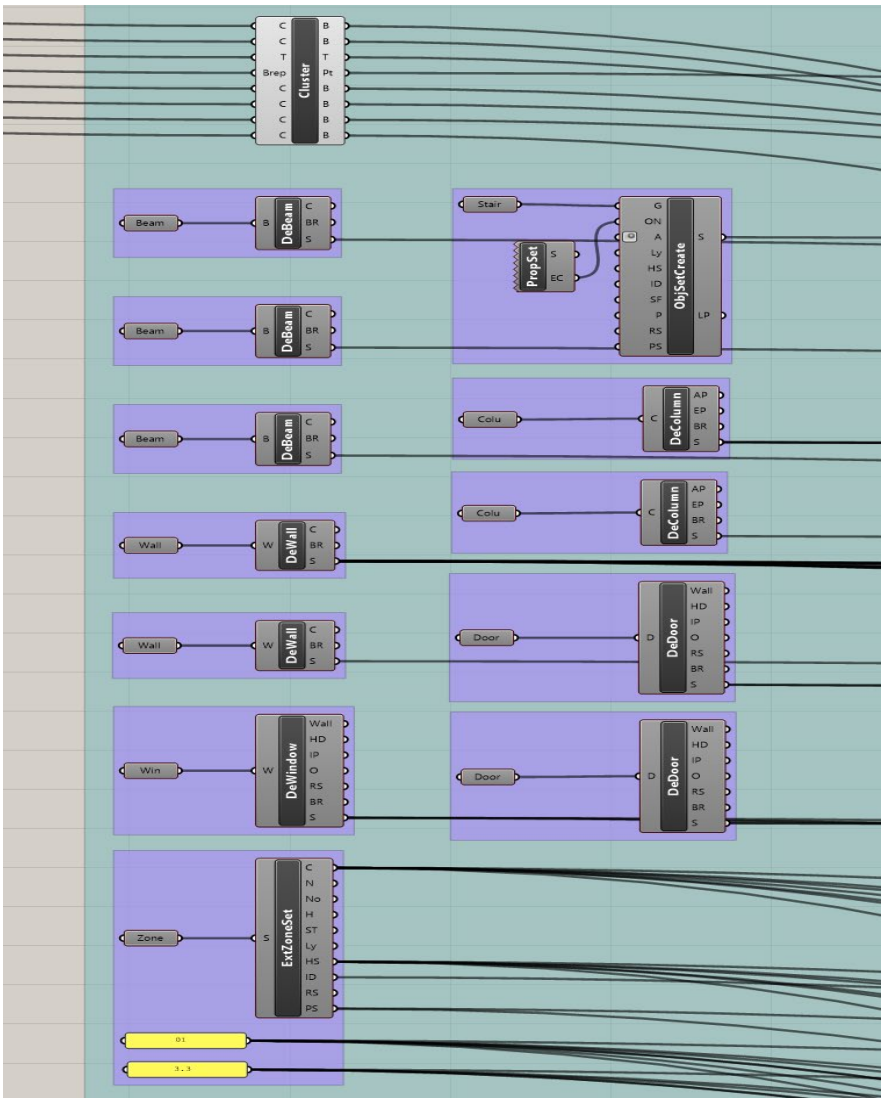
Neste sentido, a abordagem do *Design* Generativo permite que projetistas sejam liberados do processo de modelagem, uma vez que o algoritmo projetado é que irá gerar as várias soluções de projeto. No entanto, esse método fornece, geral e prioritariamente, entidades geométricas simples, sem informações semânticas da construção, insuficientes a potencialidades do BIM.

Por sua vez, sistemas de avaliação automatizada de projetos, como colocado na seção 2.2, demandam que os modelos submetidos a este tipo de avaliação possuam semântica paramétrica.

Se a proposição deste constructo prevê que no ciclo “Avaliar” haja uma análise automatizada dos indivíduos gerados, torna-se imprescindível, portanto, que a amostragem destes indivíduos não possua informações restritas à geometria da solução. Como visto até aqui, na parte (i) do algoritmo generativo, as soluções geradas são constituídas basicamente pela associação dos módulos dos ambientes, como apresentado na Figura 38.

Para tanto, para equacionar a necessidade de que a amostragem da população fosse de modelos paramétricos semânticos, foi desenvolvida a parte (ii) do algoritmo generativo, orientada à formação generativa dos modelos BIM IFC. Esta parte diz respeito ao destaque 6 do algoritmo, ilustrado na Figura 39 a seguir.

Figura 39 - Destaque 6: modelagem BIM IFC generativa



Fonte: A autora (2020).

Visto que o objetivo da geração de modelos paramétricos será a avaliação automatizada da amostragem da população (seção 4.2), é fundamental considerar que o *software* que fará este processamento demanda uma modelagem preparada para o sistema neutro de arquivos IFC[1].

[1] Ver seção 2.1.1, parágrafos 9 e 10.

Embora o Rhino-Grasshopper contenha recursos à geração de modelos BIM IFC, a saída e a visualização destes arquivos se mostram mais eficientes quando integradas a um *software* de modelagem BIM. Para esta pesquisa, foi especificado o *software* Archicad (Apêndice A) para cumprir esta integração. A comunicação entre estes sistemas ocorre através do *plug-in* “Grasshopper-Archicad Live Connection”.

O objeto da prova de conceito deste constructo prevê a geração, avaliação e evolução de um objeto arquitetônico para a fase de projeto denominada “*Early Concept Design*”, conceitualmente equivalente à fase de “Estudo Preliminar” de um projeto. A justificativa para esta escolha de fase é dada na seção 4.2 deste texto.

O CAU/BR (2014) define “Estudo Preliminar” como a “etapa de projeto destinada à concepção e à representação do conjunto de informações técnicas iniciais e aproximadas, necessários à compreensão da configuração da edificação, podendo incluir soluções alternativas.”

Isto posto, os elementos de modelagem paramétrica que constam nos modelos BIM IFC gerados foram definidos de maneira a cumprirem suficientemente o escopo formal de um Estudo Preliminar, sem a pretensão de tornarem-se Projetos Executivos. O Quadro 4 apresenta os elementos definidos para a modelagem generativa:

Quadro 4 - Elementos de modelagem Grasshopper-Archicad

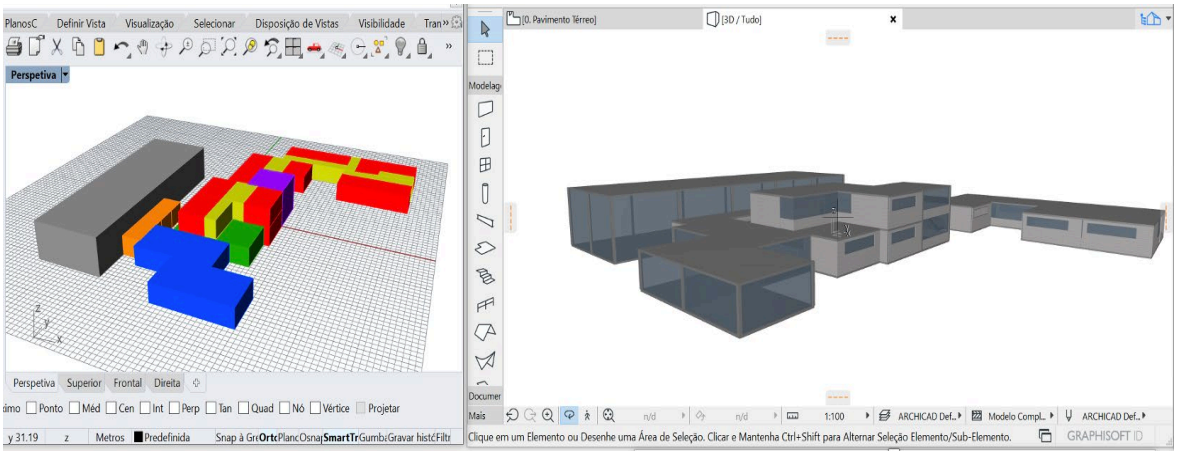
Elemento	Descrição
Beam (viga)	Gera elementos ao longo de uma ou mais linhas, polilinhas ou curvas abertas ou fechadas. Essas curvas podem estar no espaço 3D.
Column (pilar)	Requer dois pontos como parâmetros de entrada e gera pilares ao longo dos segmentos definidos (que servem como eixos do elemento).
Object (objeto / escada)	Determinados elementos de modelagem não estão disponíveis no integrador Archicad-Grasshopper, como o elemento Escada. Nestas situações estes componentes recebem inputs do Archicad como elementos do tipo “objetos da biblioteca”. Requerem, então, um único

	parâmetro de entrada: ponto de ancoragem. O tipo de peça da biblioteca padrão depende do que é definido nas configurações padrão do objeto.
Slab (laje)	O parâmetro de entrada necessário é um ou mais polígonos fechados definidos pela superfície plana dos dados do tipo curva. Como o Archicad pode interpretar apenas lajes horizontais, os polígonos de entrada devem estar no plano XY ou paralelo.
Wall (parede)	Gera elementos com base em uma ou mais linhas, polilinhas ou curvas abertas ou fechadas. As geometrias de entrada devem estar no plano XY ou paralelo.
Window / Door (janela / porta)	Ambos os elementos só podem ser colocados em paredes geradas. Portanto, os componentes “janela e porta” requerem uma entrada do tipo Parede e pontos de orientação.
Zone (zona)	Gera unidades espaciais para representar os ambientes do projeto (salas, circulação,...) e forma a base do processo de revisão do modelo. Requer duas entradas: curva que determina a forma da zona, e o ponto de localização do selo de identificação da zona.

Fonte: A autora (2020).

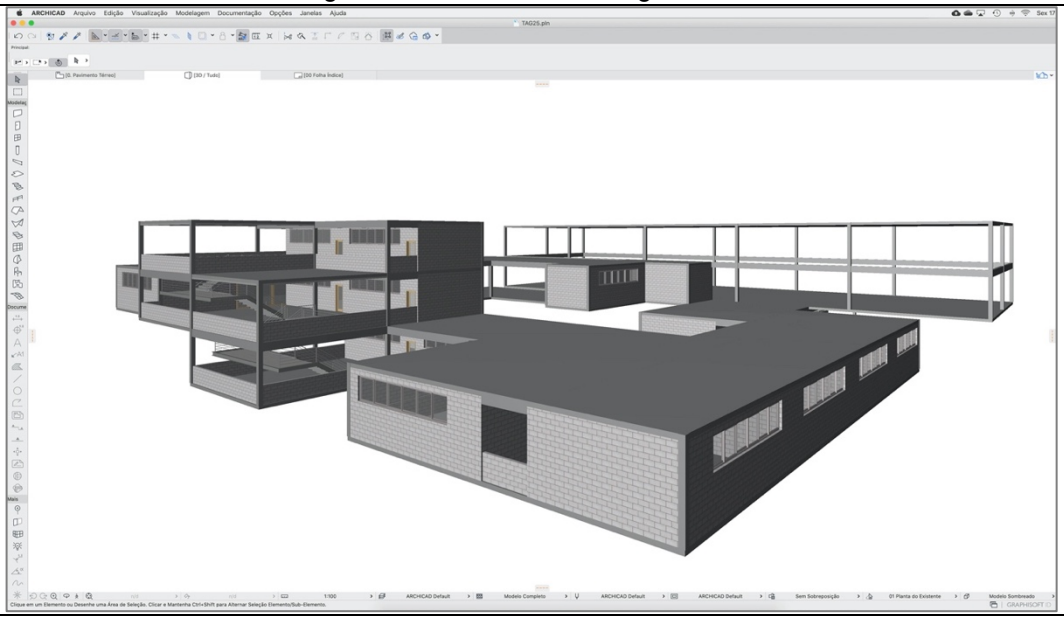
A correspondência de um indivíduo gerado no Rhino-Grasshopper *versus* Archicad é apresentada na Figura 40; e na Figura 41 vê-se um modelo BIM IFC com todos os elementos listados no Quadro 4.

Figura 40 - Modelagem generativa Rhino-Grasshopper e Archicad



Fonte: A autora (2020).

Figura 41 - Modelo BIM IFC generativo



Fonte: A autora (2020).

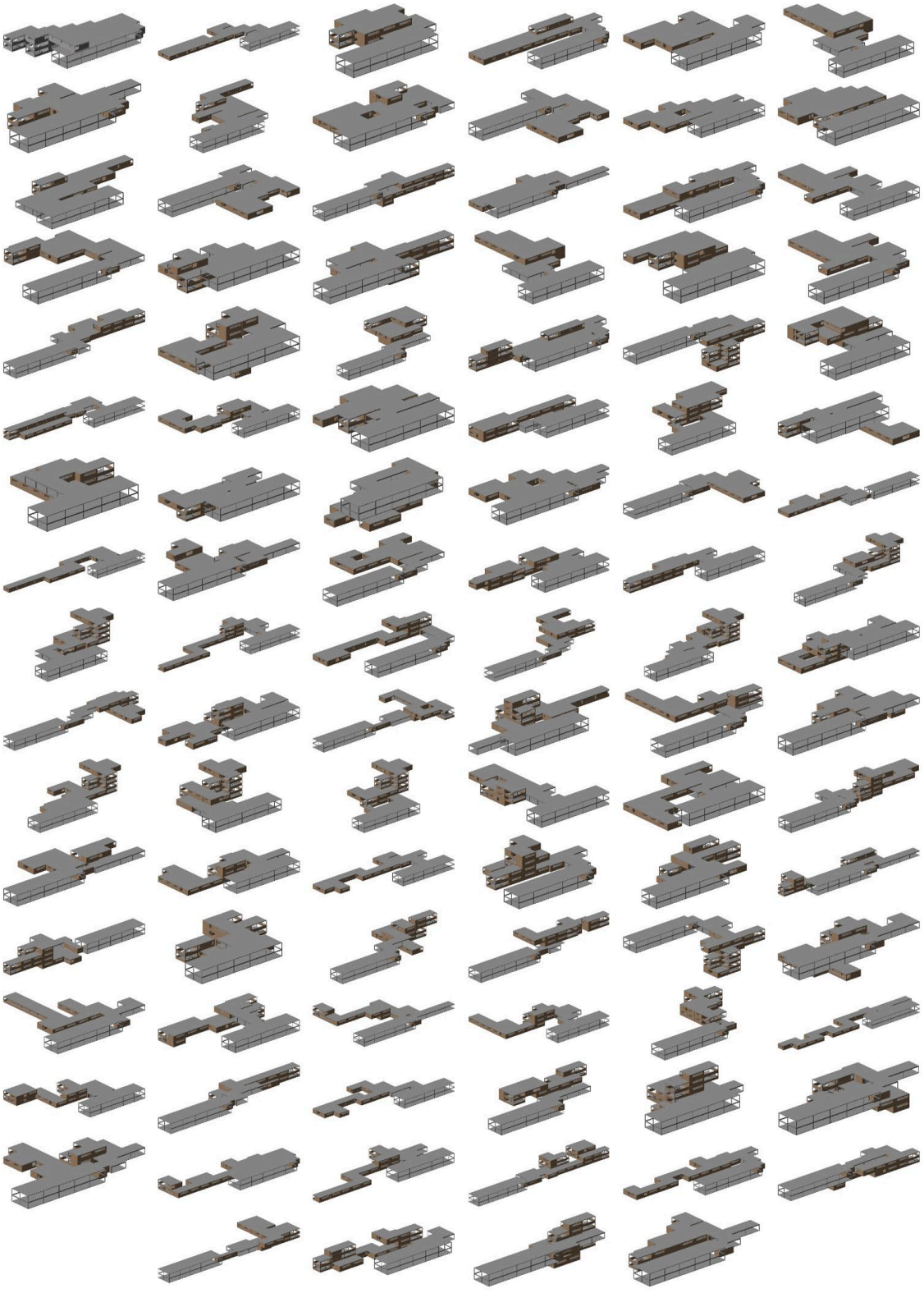
A partir da galeria de imagens da Figura 42, e no Apêndice C, é possível verificar como este processo foi aplicado aos mesmos 100 indivíduos da amostragem da população e que seguirão para os ciclos “Avaliar” e “Evoluir” do constructo.

Nesta parte do algoritmo, portanto, o Grasshopper foi utilizado como um editor de relacionamentos lógicos que definiram o modelo paramétrico. Por meio do algoritmo, foi possível estabelecer um conjunto de operações em sequência que viabilizaram o processo generativo de modelagem BIM IFC. Com a utilização do *plug-in* Grasshopper-ARCHICAD *Live Connection,* essa integração ocorreu sem troca de arquivos, tendo sido gerados de forma programada.

Cada geração de modelo BIM (um indivíduo da população) demandou o tempo médio aproximado de 13 minutos, entre abertura dos *softwares*, carregamento do arquivo do algoritmo generativo e processamento completo da modelagem no Archicad. Após a conclusão de geração de cada indivíduo, um arquivo IFC2x3 era salvo manualmente, utilizando-se o tradutor nativo “Exportação Geral” [1]. Já a exportação para o *software* SMC ocorreu de forma automática através do conector “Archicad-Solibri”.

[1] Tradutor IFC: define as regras para as quais os elementos devem ser convertidos e como devem ser interpretados – seja no ARCHICAD ou no aplicativo que lê o arquivo IFC. Exportar para fins gerais: exporta tantos elementos paramétricos com todas as propriedades e classificações.

Figura 42 - Modelagem BIM IFC generativa da amostragem da população



Fonte: A autora (2020).

4.2 Ciclo 2 - Avaliar: Verificação automatizada de regras

Em um processo computacional evolucionário, após a geração da população inicial de indivíduos, a aptidão de cada indivíduo é avaliada para selecionar quem participará da geração de novos indivíduos nas gerações seguintes, promovendo a evolução daquela solução. A Figura 43 esquematiza como ocorre o ciclo “Avaliar” do constructo.

Figura 43 - Subetapas do ciclo AVALIAR do constructo E-BIM

Fonte: A autora (2020).

Processos de avaliação são frequentemente realizados por *softwares* de análise dedicados, que podem demandar minutos, ou até horas, para avaliar as soluções da população inicial. A avaliação envolve o uso de funções (*fitness*) para atribuir pontuações de aptidão às soluções iniciais. Essas funções podem ter objetivos únicos ou múltiplos, podem ser unimodais ou multimodais, contínuos ou descontínuos, estáticas ou em constante mudança. Algoritmos genéticos são hábeis em encontrar boas soluções para todos esses tipos de funções, mas, muitas vezes, são necessárias técnicas especializadas nestes cálculos. Essa avaliação de adequação também pode ser determinada pela competição entre as soluções. Assim, classificações por pontuações (escores) também são comumente consideradas como avaliação para critérios multiobjetivos (BENTLEY, 1999).

Entretanto, problemas de aptidão e otimização relacionados a projetos de arquitetura e engenharia são complexos e definidos por muitos objetivos diferentes, nem sempre expressos em funções matemáticas aplicáveis em algoritmos ou diretamente relacionáveis somente aos parâmetros geométricos gerados em um processo de *design* generativo.

> Novos critérios surgem continuamente na arquitetura, desde os códigos de construção e segurança, até as técnicas de fabricação e montagem. Até recentemente, os únicos meios de lidar com esse complexo e crescente corpo de conhecimento eram a cognição humana e os processos de revisão organizacional. Alguns critérios envolvem análises computadorizadas (por exemplo, estruturas), mas, ainda assim, os detalhes dos fatores de segurança e a interpretação dos resultados têm sido um empreendimento manual e centrado nas pessoas. Embora algumas regras possam ser incorporadas em sistemas de geração de projeto paramétrico, a verificação automatizada de regras de projetos candidatos é apropriada quando as regras:
>
> 1. Aplicar em diferentes edifícios e sistemas espaciais, porque os objetos e regras de geração não podem ser definidos de antemão. As configurações espaciais são um exemplo.
> 2. São aqueles de agências públicas, organizações ou clientes que precisam revisar projetos que são gerados por um conjunto aberto de ferramentas de projeto.
> 3. Onde as condições sendo avaliadas são globais, tais como regras associadas à segurança, integridade estrutural, uso de energia e custos (EASTMAN *et al.*, 2009a, p.1012).

Abordado na seção 2.2 deste trabalho, o *software* SMC tem como objetivo promover a qualidade BIM, através de controle de integridade, controle de conformidade, análise de projeto e verificação de código por meio de processos de validação de interferência, espaços de circulação, segurança, códigos de construção e controle de quantitativos. Com mais de 50 modelos de conjuntos de regras nativas do *software*, os usuários podem definir regras parametrizadas para verificar seus projetos de construção. O sistema de análise baseado em regras consiste em filtragem, regras de classificação e regras de severidade. Para tal, o SMC possui um algoritmo interno que identifica e classifica a severidade dos conflitos entre: Críticas, Moderadas e Leves (LEE *et al*., 2015).

Bell *et. al.* (2009) abordam que a verificação automatizada de projetos, entretanto, é muito mais frequente na versão de um projeto completo. Entretanto, na visão dos autores, esta prática pode levar a um retrabalho caro, caso o projeto falhe na verificação. Segundo os autores, em vez de esperar que modelos BIM completos sejam submetidos à verificação, poderia-se utilizar todo o potencial do processo BIM, fazendo verificações mais cedo, quando informações relevantes estiverem disponíveis a alterações conceituais.

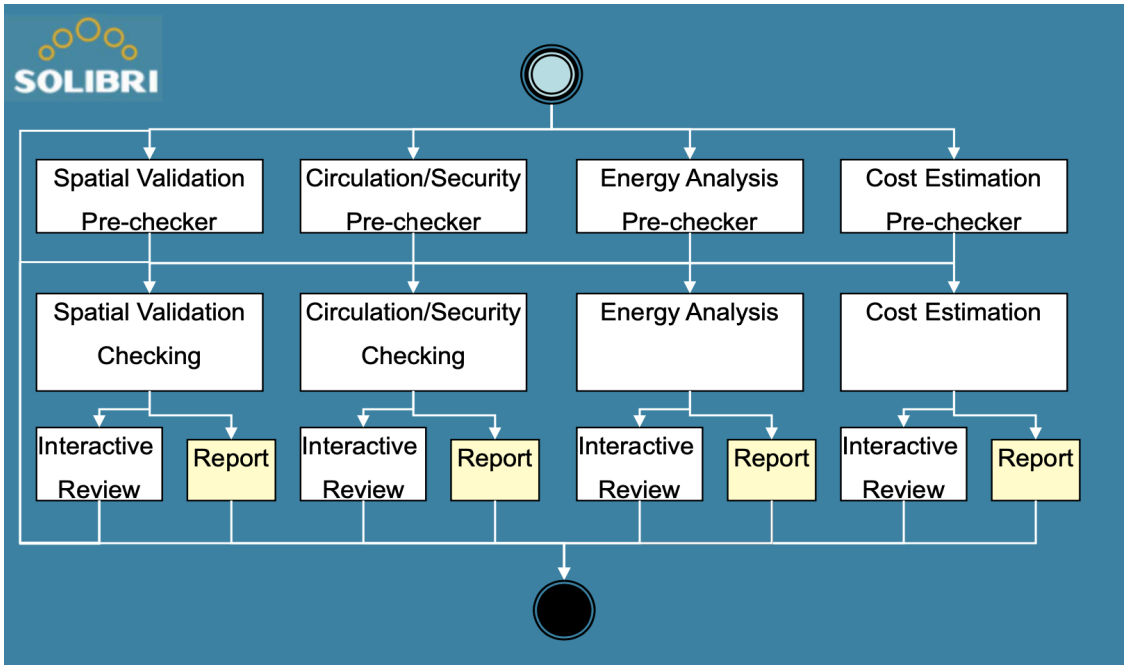
Também sob a visão da aplicabilidade das ferramentas computacionais para avaliação e otimização de performances do projeto, Eastman (2009b) afirma que o conceito inicial do projeto (*Early Concept Design*) é extremamente importante para determinar o eventual sucesso e impacto de um projeto. Para o pesquisador, – conhecido como "o pai do BIM"[1], o projeto conceitual (ou Estudo Preliminar) ainda tem sido um exercício amplamente mental de gerar vários conceitos espaciais e avaliá-los intuitivamente, com base no conhecimento e na experiência acumulada do arquiteto, como pouca assistência computacional ou lógica, nesta fase. Assim, para o pesquisador, embora a importância do Estudo Preliminar de um projeto seja reconhecida há muito tempo, as ferramentas digitais para apoiar esse estágio do processo de projeto ainda são escassas ou insuficientes.

Portanto, o BIM também pode ser uma ferramenta poderosa quando se têm requisitos programáticos complexos em andamento. Eastman (2019b) descreve, por exemplo, que sua equipe (do Laboratório de Integração AEC da Faculdade de Arquitetura do Instituto de Tecnologia da Geórgia, contratada pela Administração Geral de Serviços dos EUA - GSA) foi contratada para automatizar e analisar as diretrizes de projetos para todos os tribunais dos Estados Unidos. O objetivo era que os estudos preliminares dos arquitetos pudessem ser avaliados e verificados automaticamente de acordo com critérios específicos. O *US Courts Design Guide* possui muitos critérios relacionados à circulação, segurança e eficiência, e, assim, dentre os requisitos mínimos a serem atendidos nesta fase de concepção dos

[1] "O pai do BIM": ver página 33 e em acesso livre disponível em <https://www.rib-international.com/en/company/ blog/if-bims-a-mystery-heres-the-history/>. Acesso em 19 maio 2020.

projetos, estão o requisito “Análise de Circulação da Planta” e “Análise Preliminar de Energia”. A Figura 44 ilustra a estratégia de validação de projetos no GSA.

Figura 44 - *Early Concept Design Validation*



Fonte: Georgia Institute of Technology - Eastman (2009b).

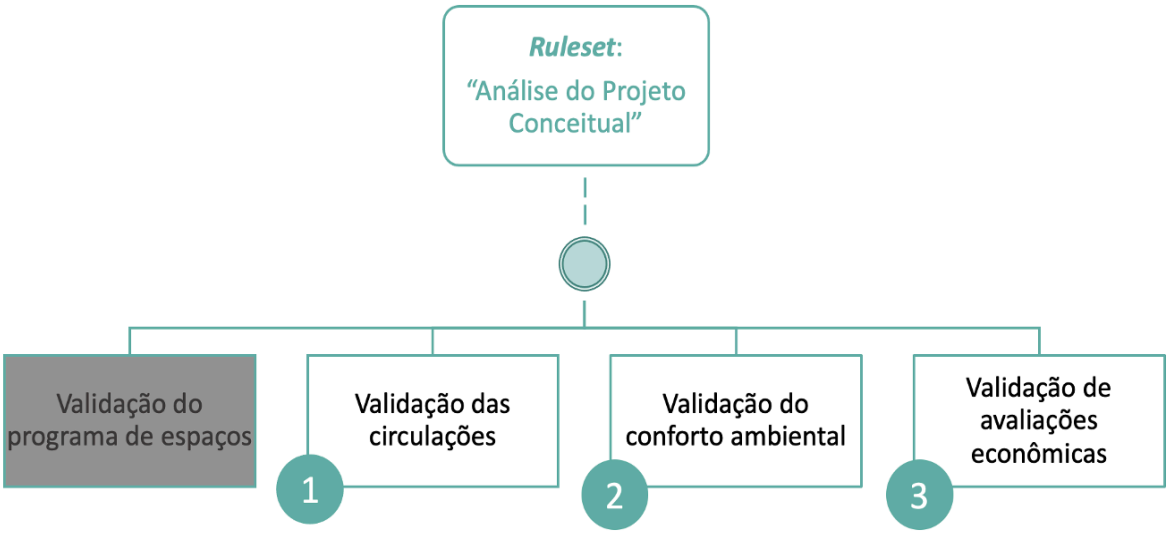
Para que as soluções de projeto do GSA sejam então analisadas, os *softwares* BIM devem proceder com a exportação de arquivos no formato neutro IFC e então lidos em APIs que trabalham em conjunto com o SMC.

4.2.1 Ruleset: “Análise do projeto conceitual”

Em analogia ao método de avaliação de projetos do GSA abordado na seção anterior, definiu-se, portanto, que as soluções geradas na população inicial fossem analisadas por regras de verificação no *software SMC*. A Figura 45 apresenta a estrutura de adaptação das estratégias de avaliação para o contexto desta pesquisa, criadas em um padrão denominado *Ruleset*: “Análise do Projeto Conceitual”, estruturada em três camadas: 1) Validação das circulações; 2) Validação do conforto ambiental; e 3) Validação de avaliações econômicas.

No SMC, um "*ruleset*" é um conjunto de regras em uma unidade funcional definida pelo usuário. Um conjunto de regras contém informações sobre: Regras do conjunto; Ordem das regras e possíveis subconjuntos de regras; Valores de parâmetros usados para as regras.

Figura 45 - Estrutura para o *ruleset*: Análise do Projeto Conceitual



Fonte: Adaptado de Georgia Institute of Technology - Eastman (2009b).

Na sequência, foram então definidos os parâmetros de avaliação que fundamentariam a criação do *ruleset*.

A seção 4.1.1 deste trabalho aborda a tipologia proposta para o objeto arquitetônico da prova de conceito do constructo (arquitetura escolar padrão FDE). Diante àquela reflexão, esclarece-se que a função formal e a estética de uma edificação são fundamentais à boa arquitetura escolar. Entretanto, é essencial lembrar também que a racionalidade e a viabilidade econômica são uma realidade determinante no que diz respeito ao investimento público. Muitas vezes, terrenos para a implantação de projetos são altamente restritivos e com características nada padronizadas. Neste âmbito, a possibilidade da variabilidade tipológica de uma edificação deve ser considerada tanto como instrumento da valorização arquitetônica quanto da otimização do investimento de recursos.

Os critérios para estas avaliações procederam de estudos técnico-científicos existentes quanto à qualidade e à economicidade do objeto arquitetônico, a saber:

- Critérios de qualidade do projeto

Pereira *et al.* (2018) apresentam em artigo um método de análise de projetos arquitetônicos escolares utilizando parâmetros de projeto, projetos de referência e comparações para o contexto brasileiro como embasamento teórico. No estudo, parâmetros de projeto foram selecionados, especificamente para a Fundação para o Desenvolvimento da Educação (FDE), responsável por gerenciar mais de 5000 escolas públicas do Estado de São Paulo, Brasil. Segundo os autores, o método contém uma estrutura de análise de projeto baseada nos parâmetros de projeto e em comparações entre os projetos precedentes e as soluções locais de projeto. É uma ferramenta de apoio ao processo de projeto para vários contextos, para inspirar uma arquitetura escolar de qualidade e com o aprendizado como foco principal. Embora ressaltem que a descrição do procedimento não é um método passo a passo, afirmam que há uma indicação ou recomendação de como proceder em uma análise de proposta de projeto usando requisitos e recomendações precedentes, conforme relaciona o Anexo C.

Destas recomendações, foram adotados os seguintes referenciais:

a) Os fluxos de circulação devem evitar becos sem saídas;
b) Os fluxos de circulação devem permitir fluxo em dois sentidos, sem obstáculos e evitar becos sem saídas;
c) Comprimento de rota de fuga da sala deve ser de até 40 m;
d) A distância de conforto de qualquer espaço às escadas deve ser até 25 m;
e) Entradas de ventilação e iluminação devem ter afastamento mínimo de 3 m de outras paredes;
f) Ambientes podem ser iluminados e ventilados indiretamente por fossos de iluminação com diâmetro mínimo de 6 m;
g) Áreas de ventilação e iluminação, em relação à área do piso, devem ter mínimos e máximos, respectivamente:

- Salas: 20% e 30%;.
- Vestiário:10% e 30%;
- Refeitório: 10% e 20%.

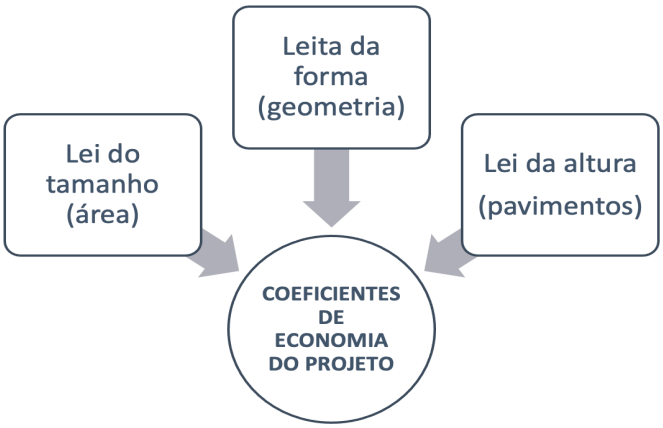
– Critérios de economicidade do projeto

Diferentes opções de solução de um programa de arquitetura produzem resultados imediatos formais e circunstanciais. Logan (1987) exemplifica como os critérios e consequências estão atrelados em um projeto: “se aumentamos o tamanho de uma janela, podemos captar mais luz e obter uma melhor vista; mas também haverá uma troca maior de calor e maiores problemas de privacidade” (*apud* FURTADO SILVA e MACIEL FURTADO SILVA, 2020).

Mascaró (2006) afirma que arquitetos são impossibilitados de controlar economicamente cada uma das decisões do projeto porque desconhecem não apenas sua influência no custo total, mas também suas inter-relações. Ao pesquisador, parece que, no caso do desempenho de custos, os arquitetos ainda necessitam de um projeto completo ou de um anteprojeto detalhado para avaliarem o impacto econômico do projeto. E, na maioria dos casos, quando pretende-se diminuir o custo verificado o caminho se dá através da diminuição da qualidade através da escolha de materiais e do tipo de execução, e, raras vezes, por meio da apropriação da forma da edificação. Em seu trabalho, o autor demonstra como cada vez que o arquiteto define uma forma arquitetônica, está determinando uma variável direta do custo de uma edificação: altura, forma da planta, relação comprimento-largura-altura do edifício, tamanho das circulações verticais e horizontais são decisões que definem os coeficientes de economia do projeto. A Figura 46 apresenta os preceitos que refletem nesses coeficientes.

Os critérios de avaliação econômica a serem adotados neste estudo são referenciados, portanto, no profundo estudo de Mascaró (2006), que analisa as relações entre a forma e a economicidade do projeto:

Figura 46 - Leis relativas à economia do projeto



Fonte: adaptado de Mascaró (2006).

a) Coeficiente de compacidade (CP): relação percentual que existe entre o perímetro de um círculo de igual área do projeto e o perímetro das paredes exteriores do projeto. Matematicamente, o índice máximo de compacidade do círculo é 100% e o do quadrado é 88,6%, e dificilmente os projetos se aproximam dele. Mas, quanto mais este índice é maximizado para se aproximar de 88,6% tanto melhor será o desempenho da solução formal em relação à compacidade.
b) Coeficiente de circulação (CR): um contrassenso pouco conhecido é que áreas de circulações têm custo de 20% a 30% superior em relação ao custo das áreas principais de edifício. A melhor estratégia para minimizar a área de uma circulação não é diminuir as larguras das circulações, mas encurtá-las. Uma possível forma de analisar a eficiência de área de circulação em um projeto com programa e parâmetros já definidos é avaliar a relação desta área e do seu comprimento total. Assim, espera-se que estas relações sejam inferiores a 25%.
c) Coeficiente de densidade de alvenarias (DS): relação entre a área de projeção das paredes e a área líquida ocupada. Da análise deste índice é possível verificar se a compartimentação do projeto foi feita de forma racional e eficiente. Quanto menor a densidade de paredes, maior a área útil do pavimento. Espera-se que este coeficiente seja inferior a 16%.
d) Coeficiente de fachada (FC): dos custos totais de um edifício, as fachadas constituem, em média, 15% a 20% do custo total do edifício. Para a economicidade da solução é importante obter o máximo de aproveitamento destas fachadas. Importante lembrar que arestas nas fachadas têm também seu

custo a ser considerado: acréscimo de 0,5m por aresta. A avaliação do coeficiente pode então ser feita pela relação entre a área da fachada e a área líquida ocupada, objetivando-se que não ultrapasse 65%.

Diante dos critérios de avaliação estabelecidos, têm-se a Figura 47, que sintetiza os parâmetros para criação do *Ruleset*: “Análise do Projeto Conceitual”.

Figura 47 - Parâmetros para o *ruleset*: “Análise do Projeto Conceitual”

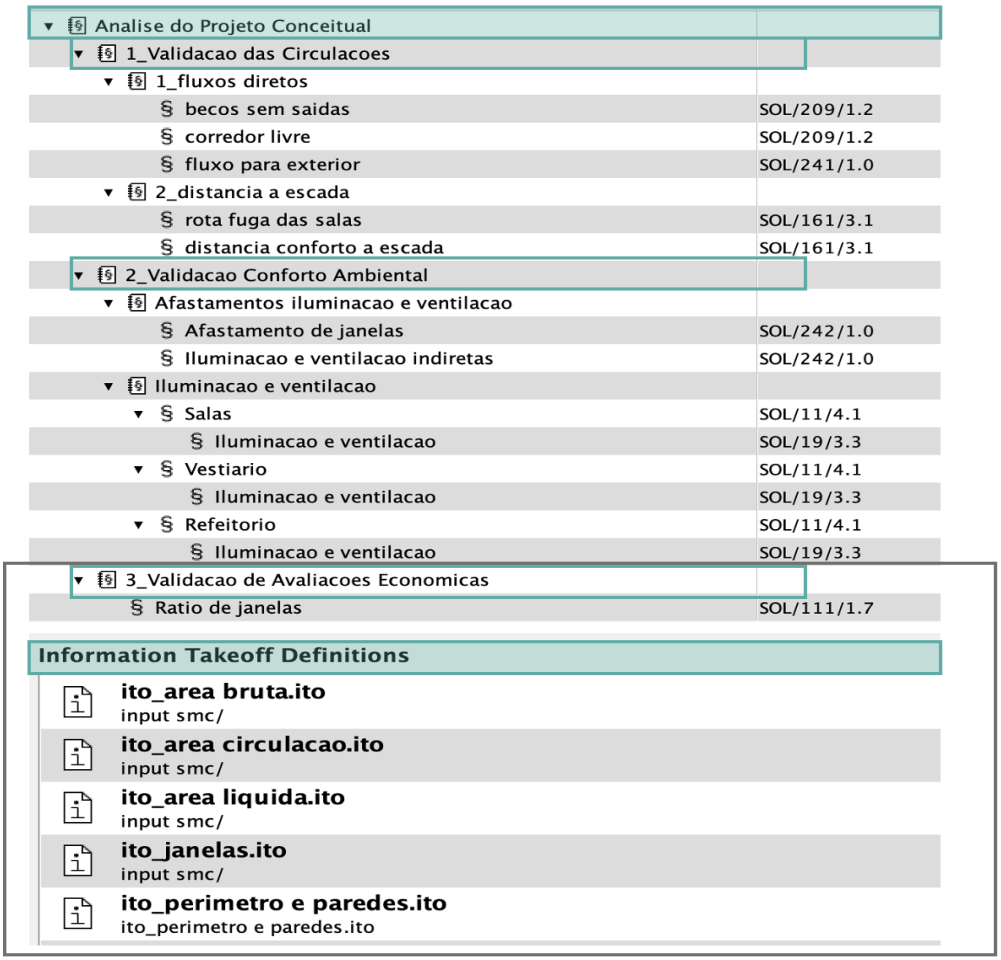
	***Ruleset* Solibri: Análise de Projeto Conceitual**	
1	**I - Validação das circulações**	
	1	Os fluxos de circulação devem evitar becos sem saídas.
	2	Os fluxos de circulação devem permitir fluxo em dois sentidos, sem obstáculos e evitar becos sem saídas.
	3	Comprimento de rota de fuga da sala deve ser de até 40 m.
	4	A distância de conforto de qualquer espaço às escadas deve ser de até 25 m.
2	**II - Validação do conforto ambiental**	
	5	Entradas de ventilação e iluminação devem ter afastamento mínimo de 3 m de outras paredes.
	6	Ambientes podem ser iluminados e ventilados indiretamente por fossos de iluminação com diâmtro mínimo de 6 m.
	7	Salas - ventilação e iluminação: mínimo de 20% e máximo de 30% da área do piso.
	8	Vestiário - ventilação e iluminação: mínimo de 10% e máximo de 30% da área do piso.
	9	Refeitório - ventilação e iluminação: mínimo de 10% e máximo de 20% da área do piso.
3	**III - Validação de avaliações econômicas**	
	10	Áreas de janelas devem ser inferiores a 15% da área de piso e inferiores a 50% da área do edifício.
	11	Coeficiente ideal de compacidade da forma deve ser 0,886.
	12	Coeficiente máximo de densidade das alvernarias deve ser 0,16.
	13	Coeficiente máximo da área de circulação deve ser 0,25.
	14	Coeficiente máximo da área de fachada deve ser 0,65.

Fonte: A autora (2020).

De acordo com Andrade e Silva (2017), o SMC contém *templates* de regras que podem ser combinados e/ou customizados, para um objetivo de análise que se pretende executar. A nomenclatura destas regras segue o padrão “SOL n”, em que se trata de um número, não sequencial, para indexar a regra. Estes *templates estão* referenciados no Anexo D, e foram utilizados na criação do *ruleset*, conforme

mostra a Figura 48, relacionando os aspectos definidos para a validação (Figura 47) com os *templates* das regras.

Figura 48 - *Ruleset*: “Análise do Projeto Conceitual”



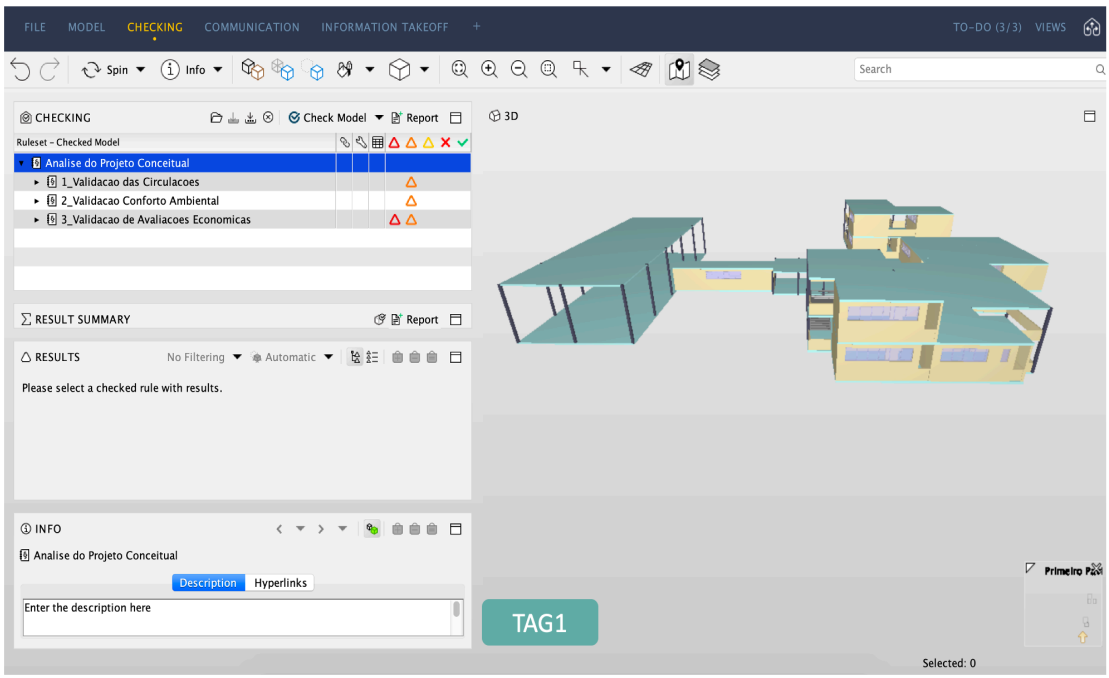
Fonte: A autora (2020).

Além de *rulesets*, o SMC também oferece o recurso de geração e personalização de relatórios do tipo *Information Takeoff* (ITO), que permite ao usuário analisar as informações dos componentes de um modelo tanto na visualização 3D quanto na forma de um relatório, que pode ser exportado para uma planilha customizada do Microsoft Excel. Essas informações incluem identificadores e classificações, locais, quantidades que podem ser totalizadas agrupando componentes juntos, bem como quaisquer valores de propriedade que estão dentro de um conjunto de propriedades. Para o incremento de dados de análise, o recurso ITO também foi atribuído ao processo de avaliação, na camada “Validação de avaliações econômicas”.

A Figura 49 apresenta a interface do SMC com um indivíduo (solução de projeto) em análise. É válido lembrar que os *softwares* de modelagem Archicad e SMC podem trabalhar com uma API de integração. Assim, ao se gerar os modelos BIM IFC no Archicad, estes foram abertos automaticamente no SMC, onde então as regras e o ITO foram executados para cada um dos 100 indivíduos da amostragem. Foram pré-configurados 5 tipos de relatórios, dos quais foram extraídas as informações do tipo ITO e as ocorrências verificadas pelas regras do *ruleset* criado. O Apêndice D contém um tipo de cada relatório extraído do SMC.

Assim, após a análise automatizada da população de amostragem, todos os resultados obtidos nos relatórios do SMC foram compilados em uma matriz única, para fins de classificação dos indivíduos mais aptos a seguirem para o processo evolutivo do ciclo "Evoluir". Como visto anteriormente, as ocorrências verificadas pelo SMC são automaticamente classificadas, podendo ser do tipo Críticas, Moderadas ou Leves. Para efeitos de totalização das ocorrências deste experimento, foram calculadas as médias ponderadas das ocorrências Críticas, Moderadas e Leves, aplicando-se pesos, de 1,0, 0,5 e 0,25, respectivamente. A Tabela 1 apresenta esta matriz de compilação dos resultados:

Figura 49 - Interface do SMC com um indivíduo em análise



Fonte: A autora (2020).

Tabela 1 - Matriz de indicadores de desempenho ciclo AVALIAR

TAG DA SOLUCAO	coeficientes				ocorrencias			ocorrências ponderadas			Total de ocorrências	
	compacidade	densidade alvenarias	circulacao	fachada	criticas	moderadas	leves	criticas	moderadas	leves		
	ideal	max.	max.	max.	ideal	ideal	ideal	pesos				
	0,8860	**0,1600**	**0,2500**	**0,6500**	**0**	**0**	**0**	**1,00**	**0,50**	**0,25**		
	0,4554	0,0772	0,4253	1,0618	24	29	8	24	15	2	28	máximo
	0,2819	0,0597	0,2859	0,3856	0	1	0	0	1	0	1	mínimo
	0,3484	0,0684	0,3615	0,8273	7,02	14,94	1,96	7,02	7,47	0,49	14,98	média
TAG0	0,3317	0,0615	0,3644	0,8188	6	14	0	6	7	0	13,00	
TAG01	0,4214	0,0729	0,3316	0,6668	6	13	4	6	7	1	13,50	
TAG02	0,3943	0,0772	0,3467	0,7561	1	22	3	1	11	1	12,75	
TAG03	0,4026	0,0688	0,3209	0,6699	0	19	2	0	10	1	10,00	
TAG04	0,4348	0,0765	0,3504	0,5822	8	14	4	8	7	1	16,00	
TAG05	0,3949	0,0704	0,2958	0,7156	12	16	3	12	8	1	20,75	
TAG06	0,3383	0,0729	0,3677	0,8396	6	20	0	6	10	0	16,00	
TAG07	0,3662	0,0673	0,3578	0,7456	2	17	1	2	9	0	10,75	
TAG08	0,3748	0,0734	0,3383	0,7484	0	18	5	0	9	1	10,25	
TAG09	0,4404	0,0759	0,2859	0,6300	9	13	3	9	7	1	16,25	
TAG10	0,4554	0,0744	0,3545	0,6686	8	16	0	8	8	0	16,00	
TAG11	0,4008	0,0705	0,3702	0,6309	5	16	3	5	8	1	13,75	
TAG12	0,4252	0,0726	0,3531	0,6670	5	16	8	5	8	2	15,00	
TAG13	0,4392	0,0759	0,2859	0,6300	9	13	3	9	7	1	16,25	
TAG14	0,3591	0,0679	0,3791	0,8574	14	9	3	14	5	1	19,25	
TAG15	0,4245	0,0743	0,3052	0,5794	8	14	4	8	7	1	16,00	
TAG16	0,3748	0,0672	0,3049	0,7104	0	15	7	0	8	2	9,25	
TAG17	0,4224	0,0733	0,3318	0,6212	4	17	1	4	9	0	12,75	
TAG18	0,3969	0,0684	0,3545	0,6848	8	23	1	8	12	0	19,75	
TAG19	0,3693	0,0674	0,3677	0,8886	0	29	6	0	15	2	16,00	
TAG20	0,3458	0,0703	0,3621	0,8557	13	11	2	13	6	1	19,00	
TAG21	0,3972	0,0711	0,2947	0,7143	10	17	3	10	9	1	19,25	
TAG22	0,3507	0,0669	0,3718	0,8039	7	15	1	7	8	0	14,75	
TAG23	0,4313	0,0733	0,3318	0,6211	4	15	3	4	8	1	12,25	
TAG24	0,4211	0,0676	0,3169	0,6848	5	18	3	5	9	1	14,75	
TAG25	0,3171	0,0636	0,3678	0,8980	3	18	1	3	9	0	12,25	
TAG26	0,3242	0,0643	0,3893	0,7878	6	20	0	6	10	0	16,00	
TAG27	0,3353	0,0615	0,3356	0,8943	2	12	2	2	6	1	8,50	
TAG28	0,3048	0,0634	0,3421	1,0588	7	13	3	7	7	1	14,25	
TAG29	0,3250	0,0699	0,4253	0,9340	7	16	1	7	8	0	15,25	
TAG30	0,3421	0,0677	0,3630	0,8651	14	11	1	14	6	0	19,75	
TAG31	0,3788	0,0703	0,3484	0,7688	5	12	3	5	6	1	11,75	
TAG32	0,3523	0,0710	0,3617	0,7645	1	23	3	1	12	1	13,25	
TAG33	0,3280	0,0663	0,3702	0,8536	11	13	1	11	7	0	17,75	
TAG34	0,3258	0,0678	0,3443	0,8226	9	13	3	9	7	1	16,25	
TAG35	0,3466	0,0735	0,3604	0,8246	12	20	1	12	10	0	22,25	
TAG36	0,3513	0,0714	0,3627	0,8680	5	27	0	5	14	0	18,50	
TAG37	0,3772	0,0706	0,3553	0,6860	3	23	1	3	12	0	14,75	
TAG38	0,3174	0,0658	0,3859	0,9429	12	10	0	12	5	0	17,00	
TAG39	0,4073	0,0737	0,3502	0,6661	9	12	6	9	6	2	16,50	
TAG40	0,3890	0,0703	0,3033	0,7214	23	8	2	23	4	1	27,50	
TAG41	0,4554	0,0751	0,3395	0,6100	11	15	3	11	8	1	19,25	
TAG42	0,4019	0,0709	0,3309	0,7420	15	12	2	15	6	1	21,50	
TAG43	0,3313	0,0627	0,3450	0,8958	7	20	0	7	10	0	17,00	
TAG44	0,3708	0,0720	0,3534	0,6189	8	15	4	8	8	1	16,50	

TAG45	0,3354	0,0673	0,3760	0,8294	0	12	4	0	6	1	7,00
TAG46	0,3436	0,0689	0,3400	0,8298	9	14	0	9	7	0	16,00
TAG47	0,3214	0,0639	0,3974	0,9198	7	12	1	7	6	0	13,25
TAG48	0,3116	0,0650	0,4145	1,0217	16	8	0	16	4	0	20,00
TAG49	0,3068	0,0641	0,4013	0,9812	10	10	1	10	5	0	15,25
TAG50	0,3135	0,0661	0,3592	0,9569	5	16	2	5	8	1	13,50
TAG51	0,3409	0,0750	0,3435	0,8378	8	24	1	8	12	0	20,25
TAG52	0,3028	0,0656	0,4196	1,0085	8	13	2	8	7	1	15,00
TAG53	0,3127	0,0645	0,3853	0,4957	9	12	0	9	6	0	15,00
TAG54	0,3373	0,0696	0,3354	0,8251	1	24	1	1	12	0	13,25
TAG55	0,3026	0,0662	0,3788	0,9065	5	12	3	5	6	1	11,75
TAG56	0,3203	0,0701	0,3755	0,3856	24	7	2	24	4	1	28,00
TAG57	0,3150	0,0662	0,3665	0,9278	3	13	2	3	7	1	10,00
TAG58	0,3201	0,0652	0,3623	0,9058	8	15	1	8	8	0	15,75
TAG59	0,3442	0,0712	0,3660	0,8258	6	21	0	6	11	0	16,50
TAG60	0,3413	0,0661	0,3913	0,9331	10	13	2	10	7	1	17,00
TAG61	0,3226	0,0637	0,4131	0,9660	7	10	0	7	5	0	12,00
TAG62	0,2970	0,0650	0,3991	0,9956	9	13	1	9	7	0	15,75
TAG63	0,3444	0,0656	0,3551	0,8579	1	22	0	1	11	0	12,00
TAG64	0,3750	0,0735	0,3354	0,7483	8	14	3	8	7	1	15,75
TAG65	0,3001	0,0629	0,4025	0,8985	4	18	0	4	9	0	13,00
TAG66	0,3788	0,0710	0,3574	0,6676	0	15	4	0	8	1	8,50
TAG67	0,3246	0,0631	0,3325	0,8208	6	16	4	6	8	1	15,00
TAG68	0,3734	0,0733	0,3595	0,7962	7	12	4	7	6	1	14,00
TAG69	0,3018	0,0654	0,4088	0,9718	6	13	2	6	7	1	13,00
TAG70	0,3087	0,0636	0,3414	0,9701	4	20	0	4	10	0	14,00
TAG71	0,3043	0,0616	0,3635	0,9686	4	28	2	4	14	1	18,50
TAG72	0,3163	0,0629	0,3621	0,9656	6	10	3	6	5	1	11,75
TAG73	0,3228	0,0638	0,3414	1,0011	9	15	0	9	8	0	16,50
TAG74	0,3121	0,0618	0,3711	0,9696	0	17	1	0	9	0	8,75
TAG75	0,3116	0,0637	0,4217	0,9714	5	12	1	5	6	0	11,25
TAG76	0,2945	0,0634	0,4186	1,0594	7	16	1	7	8	0	15,25
TAG77	0,3107	0,0689	0,3686	0,8994	3	17	2	3	9	1	12,00
TAG78	0,3441	0,0694	0,3767	0,7889	8	12	4	8	6	1	15,00
TAG79	0,3054	0,0726	0,3589	0,8947	10	12	4	10	6	1	17,00
TAG80	0,3117	0,0717	0,4140	0,9001	6	16	3	6	8	1	14,75
TAG81	0,3320	0,0697	0,3702	0,8792	5	16	2	5	8	1	13,50
TAG82	0,3103	0,0648	0,3949	0,9844	20	9	1	20	5	0	24,75
TAG83	0,3313	0,0717	0,3447	0,8673	0	1	0	0	1	0	0,50
TAG84	0,3288	0,0675	0,3930	0,9371	0	11	1	0	6	0	5,75
TAG85	0,3297	0,0676	0,3783	0,8551	15	10	3	15	5	1	20,75
TAG86	0,3762	0,0733	0,3608	0,7914	6	14	0	6	7	0	13,00
TAG87	0,3039	0,0638	0,4000	0,9365	5	12	0	5	6	0	11,00
TAG88	0,2819	0,0597	0,3598	1,0325	5	13	4	5	7	1	12,50
TAG89	0,3221	0,0730	0,4099	0,7791	21	13	0	21	7	0	27,50
TAG90	0,3374	0,0653	0,3514	0,8404	3	17	2	3	9	1	12,00
TAG91	0,3577	0,0686	0,3354	0,8263	13	13	2	13	7	1	20,00
TAG92	0,3495	0,0664	0,4119	0,8858	8	12	0	8	6	0	14,00
TAG93	0,2933	0,0610	0,4021	1,0561	3	2	1	3	1	0	4,25
TAG94	0,3518	0,0733	0,3366	0,7872	8	12	3	8	6	1	14,75
TAG95	0,3221	0,0691	0,3879	0,8989	5	15	4	5	8	1	13,50
TAG96	0,3548	0,0688	0,3393	0,7883	5	26	0	5	13	0	18,00
TAG97	0,2914	0,0602	0,3491	1,0618	3	13	2	3	7	1	10,00
TAG98	0,2953	0,0620	0,4128	0,9623	0	14	0	0	7	0	7,00
TAG99	0,3367	0,0767	0,3325	0,8434	11	11	2	11	6	1	17,00

Fonte: A autora (2020).

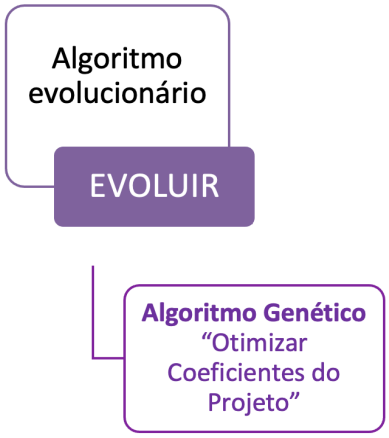
4.3 Ciclo 3 - Evoluir: otimização por algoritmo evolucionário

Como visto na seção 2.4 deste trabalho, algoritmos baseados em metaheurísticas são a estratégia dominante para resolver problemas de otimização em projetos generativos, pois funcionam através da amostragem direta do espaço de busca e se mostraram muito eficazes em evitar condições de ótimo local e espaços descontínuos de aprendizagem. Conforme fundamentam as seções 2.4 e 2.4.1 deste trabalho, os Algoritmos Genéticos demonstram ser apropriados a problemas de otimização de *design*, como o desta pesquisa.

Então, definido o espaço de busca de otimização do *design* (seção 4.1) e estabelecidas as métricas para avaliá-lo (seção 4.2), a pesquisa tornou-se apta a receber o incremento do algoritmo de otimização, para exploração das soluções de alto desempenho, conforme propõe a ideia do *Design* Evolucionário (seção 2.5).

Assim, no ciclo "Evoluir", conforme esquematiza a Figura 50, é então implementando o algoritmo evolucionário genético, otimizando a qualidade do processo generativo, em que as novas gerações de populações de soluções contenham características superiores, ou computacionalmente evoluídas, em relação à geração inicial do *design* generativo, conforme os indicadores estabelecidos.

Figura 50 - Subetapas do ciclo EVOLUIR do constructo E-BIM



Fonte: A autora (2020).

O *software* Grasshopper, utilizado para o *design* generativo da população inicial, oferece um componente evolutivo denominado Galapagos, que propõe simplificar a implementação dos algoritmos evolucionários, pois o *script* algorítmico faz parte da sua estrutura interna, não sendo necessário ao usuário programar o algoritmo. Entretanto, mesmo oferecendo algumas opções de configurações que possibilitam certa flexibilidade na modelagem do algoritmo evolucionário, o componente apresenta limitações, conforme descreve seu desenvolvedor:

> O Galapagos ainda é um produto muito jovem e realmente não teve tempo de se posicionar firmemente em qualquer fluxo de trabalho. Parece ser capaz de resolver problemas relativamente pequenos rapidamente, mas certamente precisa de muito trabalho para torná-lo mais robusto e utilizável. É provável que as aplicações mais eficazes para um solucionador desse tipo e capacidade sejam problemas pequenos ou parciais. Tentar evoluir qualquer coisa complicada quase certamente resultará em frustração (RUTTEN, 2010, s.p.).

Nagy (2017c) descreve que a partir da versão 5 o Rhino-Grasshopper possui a linguagem de programação Python incorporada a ele. Além de executar qualquer código Python padrão, ele também pode se comunicar com a interface Rhino-Grasshopper e trabalhar com geometria usando um conjunto de bibliotecas Python. A interface é feita através de um “nó Python” no Grasshopper, que permite incorporar código aos seus modelos. O código gravado nesses nós usa as mesmas bibliotecas para manipular geometria e executar várias tarefas de modelagem e é capaz de se comunicar com os ambientes Rhino-Grasshopper. Esta aplicabilidade permite misturar e combinar o código com os nós tradicionais do Grasshopper. Assim, não há exigência em desenvolver todo o modelo apenas através do código começando com um arquivo de texto em branco, e, em vez disso, pode-se desenvolver a maior parte do modelo usando nós do Grasshopper e usar apenas nós Python para executar tarefas mais complexas que são difíceis ou impossíveis no Grasshopper padrão.

Embora não seja necessário trabalhar totalmente em linguagem de programação neste caso, para muitos autores, a linguagem Python é relativamente simples de ser incorporada e, portanto, bem assimilável também para *stakeholders* do setor

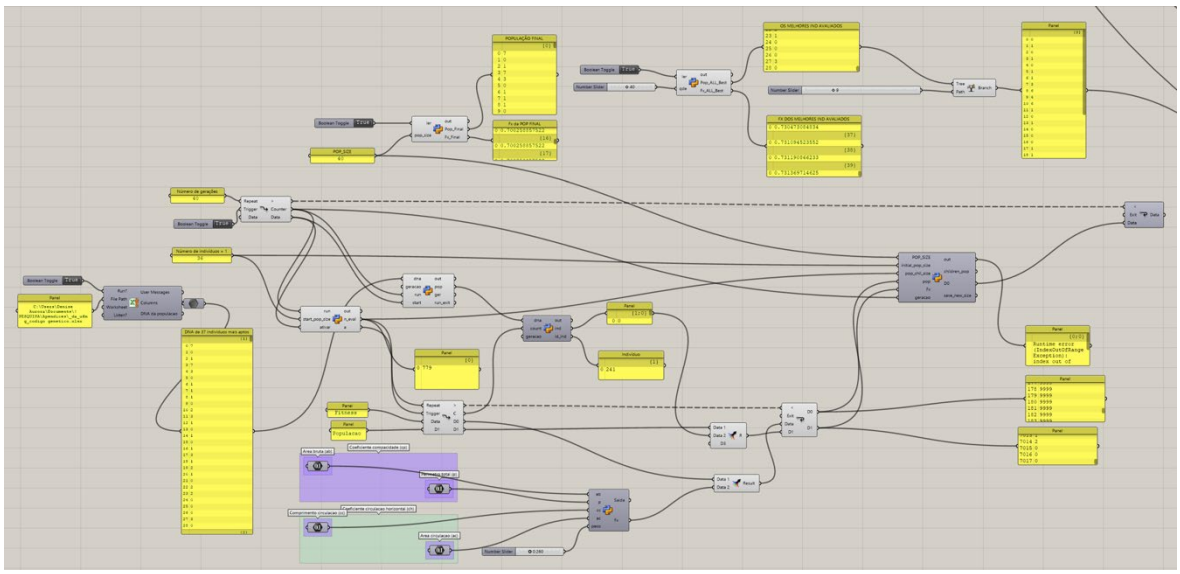
AEC. Essa popularidade pôde ser atribuída ao Python pela: (I) sintaxe relativamente simples, com foco na simplicidade, clareza e legibilidade, o que torna menos complicado aprender e escrever; (ii) extensibilidade através de uma grande coleção de bibliotecas externas; e (iii) comunidade de suporte de usuários ativos. (NAGY, 2017c).

4.3.1 Algoritmo genético "Otimizar coeficientes do projeto"

- Destaque 7: implementação do algoritmo genético no *design* generativo

Diante a este contexto, a implementação do algoritmo evolucionário ocorreu em dois momentos: na "Fase A", com o desenvolvimento de um *script* Python, e na "Fase B - ciclo "Evoluir", com a inserção no fluxo do algoritmo generativo. A Figura 51 representa o destaque 7 do algoritmo generativo, localizando essa implementação do algoritmo genético.

Figura 51 - Destaque 7: Implementação do algoritmo genético



Fonte: A autora (2020).

A codificação de um AG em Python, ou demais linguagens, pode partir de uma estrutura existente, conforme demonstra o Anexo E, ou ser totalmente reescrita. Pelas caraterísticas do componente GHPython foi necessário construir o AG em

componentes separados e criar um mecanismo de comunicação e manipulação de arquivos de registros em um diretório específico.

Os genes, com os quais trabalha um AG, são compreendidos como entidades que correspondem às variáveis do problema em estudo. Ao aplicar um algoritmo genético, é importante entender que todas as operações de otimização acontecem no nível do genótipo, enquanto a seleção é baseada no desempenho real em um determinado ambiente, ou seja, no nível do fenótipo. Os conceitos relacionados ao mapeamento genótipo-fenótipo estão abordados na seção 2.4.1 deste trabalho.

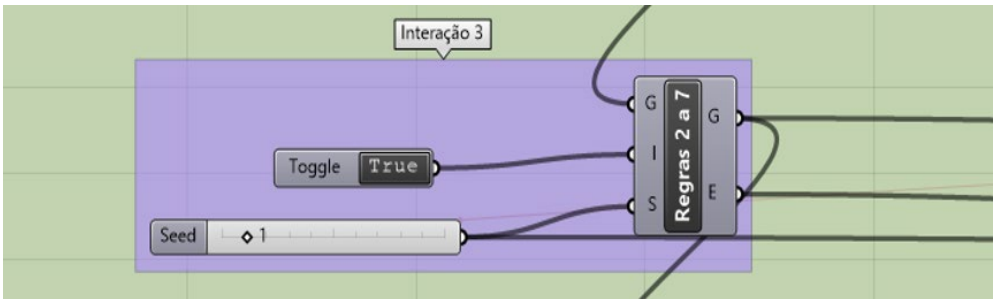
Conforme Duarte *et al.* (2013), ao aplicar um algoritmo genético, certos elementos devem ser considerados:

- O fenótipo deve ser especificado, ou seja, as soluções permitidas para o problema devem ser definidas pela especificação e enumeração de um espaço de busca;
- O genótipo (ou codificação das soluções permitidas) deve ser definido;
- A função de aptidão deve ser criada, a fim de permitir a avaliação de possíveis soluções do problema para o AG.

Tendo sido o campo de busca dos fenótipos definido na seção 4.2 desta pesquisa (Figura 38), ciclo “Avaliar”, o passo seguinte foi definir uma representação adequada para o mapeamento genótipo-fenótipo correspondente.

Na seção 4.1.2 foi visto que cada solução possível gerada para a arquitetura escolar – objeto da prova de conceito – foi tratada como um indivíduo de uma população. Nos destaques 2, 3 e 4 do algoritmo generativo (figuras 31, 32 e 33, respectivamente), tem-se 29 interações possíveis, através dos *inputs* dos componentes “*seed*” do Grasshopper. A Figura 52, a seguir, recapitula esse componente que é alimentado por um domínio de números naturais.

Figura 52 - Componente “*seed*”: *input* de interação das regras



Fonte: A autora (2020).

É visto que no algoritmo generativo, para o atendimento às regras da Gramática da Forma, cada indivíduo válido possui uma sequência de caracteres numéricos, denominada *string*. Como tratado na seção 2.4.1 deste texto, para efeitos de mapeamento genótipo-fenótipo, pode-se comparar esta *string* aos códigos cromossômicos, ou DNA, de um indivíduo.

O Quadro 5 apresenta os domínios de *inputs* por posições que compõem a *string* do DNA de cada indivíduo. As iterações dizem respeito às entradas do algoritmo para composição da Gramática da Forma dos blocos B1 a B3 (seção 4.1.1) e das conexões entre estes blocos. O domínio de cada posição da *string* é formado por números naturais definidos em cada “*seed*” do algoritmo, variando conforme a estruturação da regra dentro de cada *cluster*.

Quadro 5 - Domínios de composição dos DNAs

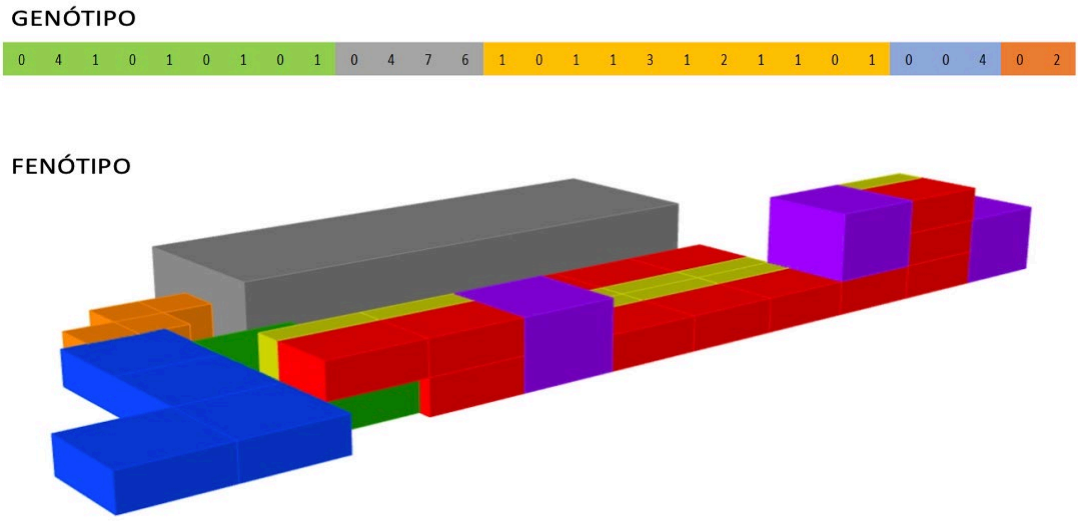
Interações	Posições	Domínio
B1	1	N={0, 1, ..., 9}
	2	N={0, 1, ..., 9}
	3	N={0, 1, ..., 9}
	4	N={0, 1, ..., 9}
	5	N={0, 1, ..., 9}
	6	N={0, 1, ..., 9}
	7	N={0, 1, ..., 9}
	8	N={0, 1, ..., 9}
	9	N={0, 1, ..., 9}
B2	10	N={0, 1, ..., 9}
	11	N={0, 1, ..., 9}
	12	N={0, 1, ..., 9}
	13	N={0, 1, ..., 9}

B3	14	N={0, 1, 2, 3}
	15	N={0, 1, 2, 3}
	16	N={0, 1, 2, 3}
	17	N={0, 1, 2, 3}
	18	N={0, 1, 2, 3}
	19	N={0, 1, 2, 3}
	20	N={0, 1, 2, 3}
	21	N={0, 1, 2, 3}
	22	N={0, 1, 2, 3}
	23	N={0, 1, 2, 3}
	24	N={0, 1, 2, 3}
Conexões	25	N={0, 1, ..., 9}
	26	N={0, 1, 2, 3}
	27	N={0, 1, 2, 3, 4}
Conexões	28	N={0, 1, 2, 3, 4}
	29	N={0, 1, 2, 3}

Fonte: A autora (2020).

Têm-se estabelecido, portanto, a correlação entre genótipo e fenótipo dos indivíduos do algoritmo evolucionário E-BIM, como ilustra a Figura 53.

Figura 53 - Genótipo e fenótipo de um indivíduo



Fonte: A autora (2020).

É válido lembrar que na seção 4.1.2 foi abordada a nomeação de cada indivíduo pelo padrão “TAG n”. Assim, cada genótipo mapeado relaciona-se a seu fenótipo por nome próprio, garantindo a busca destes dados com precisão para as operações genéticas no AG, posteriormente. A Figura 54 ilustra este padrão:

Figura 54 - Fenótipos e genótipos de indivíduos da população

Fenótipo	Genótipo																											
TAG0	7	0	1	7	3	0	1	1	1	0	2	3	1	0	1	0	1	3	1	2	1	0	2	2	0	0	0	3	0
TAG01	0	1	0	1	0	1	1	3	6	2	0	7	2	0	1	0	1	3	1	1	1	0	2	2	0	0	0	1	0
TAG02	1	4	0	1	1	3	1	0	3	4	6	6	2	1	0	0	1	3	0	0	0	1	0	1	0	0	0	0	0
TAG03	1	4	0	1	1	3	1	0	3	4	6	6	2	1	0	0	1	3	0	0	0	1	0	1	0	0	0	0	0
TAG04	0	4	1	0	1	0	1	0	1	0	4	7	6	1	0	1	1	3	1	2	1	1	0	1	0	0	4	0	2
TAG05	0	1	1	0	3	1	0	1	0	4	0	2	2	0	1	1	1	0	0	1	1	1	0	0	1	0	0	2	1
TAG06	0	1	1	0	3	1	0	1	0	4	0	2	2	0	1	1	1	0	0	1	1	1	0	0	1	0	0	2	1
TAG07	1	1	0	4	1	6	3	3	0	4	6	3	6	1	0	0	0	1	1	0	1	0	0	1	0	0	0	1	0
TAG08	0	1	1	0	3	1	0	1	0	4	0	2	2	0	1	1	1	0	0	1	1	0	0	0	1	0	0	2	1

Fonte: A autora (2020).

Como visto nas seções 2.4.1 e 2.5, problemas de otimização de *design* normalmente são complexos por conterem múltiplos objetivos, muitas vezes, conflitantes entre si. Nessas situações, a estratégia de definição da(s) função (funções) de aptidão (*fitness*) torna-se fundamental para a viabilidade de implementação do sistema de otimização, uma vez que a complexidade computacional[1] de execução do algoritmo pode comprometer a resposta ao problema. Assim, objetivando manter viável o experimento, a pesquisa foi orientada a optar pela "articulação *a priori*", em que os objetivos são convertidos em um único item na operação de otimização, e implementa-se um AG mono-objetivo.

Primeiramente, também a favor da viabilização computacional do experimento, dos critérios de avaliação utilizados no ciclo "Avaliar" (seção 4.2.1), foram selecionados dois deles, como os objetivos de otimização no processo evolutivo:

a) **Coeficiente de compacidade**: maximizar para aproximar de 88,6% (0,866);
b) **Coeficiente de circulação**: minimizar do máximo ideal de 25% (0,25), tolerando-se, com depreciação, soluções de até 50% (0,5).

A seção 2.4.1 discorre sobre a estratégia de "articulação *a priori*". Neste conceito, os dois objetivos de otimização do problema foram combinados em uma única função *fitness*, como representado na Figura 55:

[1] A teoria da complexidade computacional é um ramo da teoria da ciência da computação que se concentra em classificar problemas computacionais de acordo com suas dificuldades inerentes como disponibilidade de infraestrutura, tempo de processamento e gasto energético (DE SIQUEIRA, 2019).

Figura 55 - Função *fitness* da "articulação *a priori*" do experimento

- Coeficiente de compacidade: cp
- Coeficiente circulação horizontal: ch
- Peso: λ

Função objetivo:

$$\min \quad \lambda F_{cp} + (1-\lambda) F_{ch}$$

onde

$$F_{cp} = \frac{|0.86 - cp|}{0.86}$$

e

$$F_{ch} = \left\{ \begin{array}{ll} 0, & ch \leq 0.25 \\ \frac{ch}{0.5}, & 0.25 < ch \leq 0.5 \\ 1, & ch > 0.5 \end{array} \right\}.$$

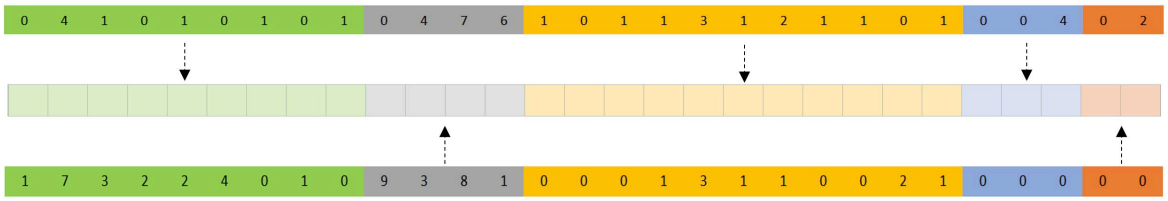
Fonte: A autora (2020).

Como método de seleção do AG foi adotada a estratégia de seleção elitista (seção 2.4.1), na qual os melhores indivíduos permanecem e formam a futura geração. Arbitrariamente, pretendeu-se selecionar os 30 melhores indivíduos. Todavia, por análise estatística (seção 5.1), houve empate dos 36 primeiros indivíduos, que então vieram participar da reprodução. O algoritmo processou num espaço de busca de 40 gerações, que teve um tempo de processamento aproximado de 8 horas.

Para a reprodução dos indivíduos – processo através do qual se combinam os cromossomos dos pais para formar novos filhos, as operações genéticas empregadas foram o cruzamento (*crossover*) e a mutação, dadas as particularidades do problema:

- *Crossover*: a estratégia de cruzamento adotada selecionou aleatoriamente segmentos entre dois indivíduos. Cada segmento foi definido com uma das cinco sequências do cromossomo que formam um indivíduo (TAG). Essa dinâmica foi adotada a fim de se manter alguma similaridade com os pais, preservando características originais bem avaliadas. Cada par de indivíduos gerou um único filho (*offspring)*. A Figura 56 demonstra um possível cruzamento.

Figura 56 - Estratégia de *crossover* da população



Fonte: A autora (2020).

Todos os indivíduos de uma população participaram do processo de *crossover*, que ocorreu entre todos os possíveis pares daquela população. Assim, a cada geração foram avaliados 630 filhos, dados por N_{off} , onde:

Figura 57 - Cálculo de indivíduos avaliados por geração

$$N_{off} = \frac{(N_{pop} \times (N_{pop} - 1))}{2}$$
$$\text{e } N_{pop} \text{ é o tamanho da população;}$$

Fonte: A autora (2020).

- Mutação: a alteração de uma única posição do código cromossômico pode mudar totalmente um indivíduo deste problema (dada a estruturação generativa), podendo, inclusive, até torná-lo inviável. Assim, quando ocorria a mutação (processo aleatório AG), eram alteradas todas as posições do DNA – respeitando os domínios do Quadro 5, possibilitando, assim, a introdução de nova diversidade ao processo evolutivo.

Ao final do processamento do algoritmo genético, para fins de comparação e análise do constructo, repetiu-se todo o fluxo do ciclo “Avaliar” para obtenção e comparação dos novos resultados. Desta vez, foram avaliados 36 novos indivíduos. Esta população de filhos (*offspring*), para melhor análise estatística, corresponde à mesma quantidade de indivíduos classificados anteriormente. A Tabela 2 apresenta, portanto, os indicadores de desempenho dos filhos do ciclo “Evoluir”.

Tabela 2 - Matriz de indicadores de desempenho ciclo EVOLUIR

TAG DA SOLUCAO	coeficientes				ocorrencias			ocorrências ponderadas			Total de ocorrências	
	compacidade	densidade alvenarias	circulacao	fachada	criticas	moderadas	leves	criticas	moderadas	leves		
	ideal	max.	max.	max.	ideal	ideal	ideal	pesos				
	0,8860	**0,1600**	**0,2500**	**0,6500**	**0**	**0**	**0**	**1,00**	**0,50**	**0,25**		
	0,5080	0,0808	0,3624	0,6750	5	23	6	5	12	2	14	máximo
	0,4636	0,0664	0,2745	0,5176	0	9	0	0	5	0	7	mínimo
	0,4826	0,0747	0,3162	0,5920	1,41	16,05	2,05	1,41	8,03	0,51	9,95	média
TAG0	0,5080	0,0740	0,3090	0,5824	0	19	0	0	10	0	9,50	
TAG01	0,5080	0,0724	0,2745	0,5824	0	17	0	0	9	0	8,50	
TAG02	0,5065	0,0748	0,3012	0,5824	3	12	0	3	6	0	9,00	
TAG03	0,5065	0,0756	0,2902	0,5824	3	16	0	3	8	0	11,00	
TAG04	0,5065	0,0765	0,3357	0,5824	3	14	0	3	7	0	10,00	
TAG05	0,5051	0,0765	0,3357	0,5824	3	19	0	3	10	0	12,50	
TAG06	0,4978	0,0774	0,3169	0,5950	0	19	0	0	10	0	9,50	
TAG07	0,4978	0,0683	0,3169	0,5950	0	20	0	0	10	0	10,00	
TAG08	0,4964	0,0776	0,2980	0,5950	3	16	0	3	8	0	11,00	
TAG09	0,4943	0,0750	0,3090	0,5887	2	19	4	2	10	1	12,50	
TAG10	0,4911	0,0740	0,3090	0,5824	0	19	0	0	10	0	9,50	
TAG11	0,4911	0,0724	0,2745	0,5824	0	17	0	0	9	0	8,50	
TAG12	0,4908	0,0733	0,2745	0,5887	2	13	4	2	7	1	9,50	
TAG13	0,4908	0,0750	0,3090	0,5887	2	15	4	2	8	1	10,50	
TAG14	0,4880	0,0808	0,3624	0,5887	4	14	4	4	7	1	12,00	
TAG15	0,4870	0,0741	0,3357	0,5392	2	17	0	2	9	0	10,50	
TAG16	0,4867	0,0802	0,3012	0,5176	2	23	2	2	12	1	14,00	
TAG17	0,4839	0,0664	0,3090	0,6039	0	19	4	0	10	1	10,50	
TAG18	0,4799	0,0779	0,3357	0,6255	1	18	0	1	9	0	10,00	
TAG19	0,4766	0,0752	0,3090	0,5887	2	17	2	2	9	1	11,00	
TAG20	0,4766	0,0757	0,3467	0,5887	2	14	6	2	7	2	10,50	
TAG21	0,4753	0,0733	0,3357	0,6750	0	15	4	0	8	1	8,50	
TAG22	0,4737	0,0750	0,3545	0,5887	4	11	4	4	6	1	10,50	
TAG23	0,4695	0,0733	0,3278	0,6318	0	14	6	0	7	2	8,50	
TAG24	0,4686	0,0740	0,3090	0,6039	0	23	0	0	12	0	11,50	
TAG25	0,4683	0,0750	0,3357	0,6318	2	11	4	2	6	1	8,50	
TAG26	0,4683	0,0742	0,3278	0,6318	0	17	6	0	9	2	10,00	
TAG27	0,4683	0,0751	0,3545	0,5456	0	15	6	0	8	2	9,00	
TAG28	0,4683	0,0733	0,2980	0,6318	2	9	4	2	5	1	7,50	
TAG29	0,4683	0,0789	0,3090	0,5887	2	15	4	2	8	1	10,50	
TAG30	0,4670	0,0746	0,3357	0,6318	0	11	4	0	6	1	6,50	
TAG31	0,4670	0,0784	0,3545	0,5887	5	10	4	5	5	1	11,00	
TAG32	0,4661	0,0724	0,2745	0,5824	0	17	0	0	9	0	8,50	
TAG33	0,4648	0,0724	0,2745	0,5824	0	17	0	0	9	0	8,50	
TAG34	0,4648	0,0740	0,3090	0,5824	0	19	0	0	10	0	9,50	
TAG35	0,4642	0,0714	0,3090	0,5608	0	19	0	0	10	0	9,50	

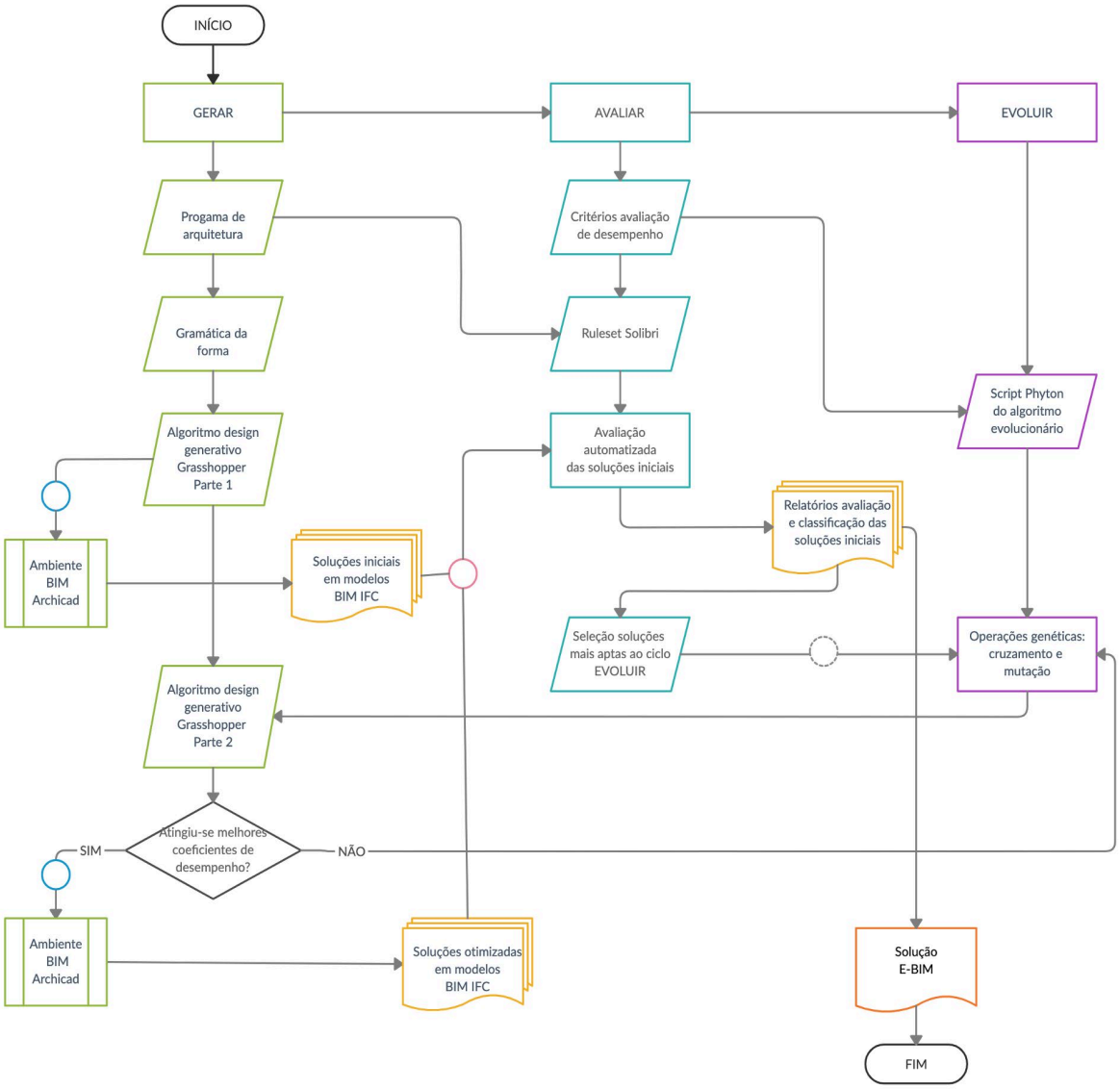
Fonte: A autora (2020).

Embora a nomenclatura para os indivíduos mantenha o padrão “TAG n”, cada indivíduo neste ciclo “Evoluir” é um novo indivíduo e não tem correspondência ao homônimo do ciclo “Avaliar”.

5 RESULTADOS E DISCUSSÃO

Este trabalho apresenta resultados da integração das tecnologias BIM, *design* generativo, verificação automatizada de projetos e algoritmos evolucionários, demonstrando contribuições para o incremento da geração otimizada de soluções para projetos de edificações – em especial na fase do *Early Concept Design* –, evidenciando, assim, avanços na aplicabilidade da inteligência artificial no campo de projetos da construção civil. A Figura 58 representa a síntese do **constructo E-BIM**, conforme os procedimentos realizados na pesquisa, detalhados no capítulo 4.

Figura 58 - Síntese do constructo E-BIM



Fonte: A autora (2020).

Na figura 58, as simbologias de círculos demonstram conexões automatizadas entre os *softwares* através de APIs já existentes, citadas na seção 4.1.3. Já para a coleta de dados de avaliação, conforme menciona a seção 4.2, foram criados *templates* para os relatórios do SMC, que se ligaram automaticamente com a planilha de compilação do *software* Microsoft Excel. O círculo tracejado deste processo também está representado na Figura 58.

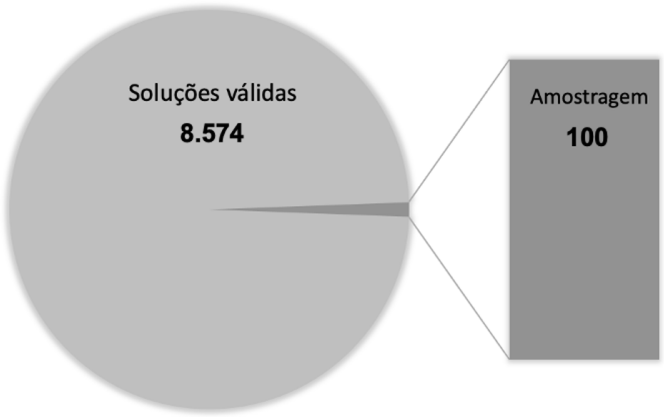
O processo desenvolvido - não nativo aos *softwares* utilizados, permitiu a automatização do processamento de 500 relatórios do Ciclo 2 - “Avaliar” e 185 relatórios do Ciclo 3 - “Evoluir”, viabilizando a interoperabilidade entre os sistemas.

5.1 Resultados quantitativos

- Resultados do “Ciclo 1 - Gerar”

A seção 4.1 deste trabalho discorre sobre o processo generativo na criação das soluções iniciais do objeto de prova de conceito (Arquitetura Escolar), por meio de um algoritmo desenvolvido nos aplicativos Rhinoceros-Grasshopper. Neste “Ciclo 1” do constructo, foram geradas **8.574 soluções válidas**, conforme o tempo estabelecido para o experimento. Para continuidade ao “Ciclo 2” seguinte, desta população de soluções válidas, foi retirada uma amostragem de **100 indivíduos**. O Apêndice B apresenta os indivíduos desta amostragem.

Figura 59 - Amostragem das soluções válidas



Fonte: A autora (2020).

Seguindo o processo generativo, com a amostragem de indivíduos, foram gerados automaticamente **100 modelos BIM IFC**, como verifica-se no Apêndice C. Conforme descreve a seção 4.1.3, o tempo total para a geração destes modelos foi de, aproximadamente, 1.300 minutos. Assim, observa-se que após o desenvolvimento e implementação do algoritmo generativo, **foi possível, em menos de 24h**, **gerar 100 modelos BIM IFC válidos para o objeto arquitetônico escolar, em fase de "*Early Concecpt Design*".**

- Resultados do "Ciclo 2 - Avaliar"

A seção 4.2 apresenta os processos onde foram estabelecidos indicadores de medição de desempenho dos indivíduos da amostragem e a coleta de dados quantitativos a respeito destes. O procedimento deste "Ciclo 2" permitiu ainda a classificação destes indivíduos, formando os candidatos ao processo evolucionário do "Ciclo 3" seguinte.

Assim, para esta avaliação, foi elaborada uma análise estatística[1] exploratória das variáveis obtidas com os indicadores estabelecidos na seção 4.2.1. (Tabela 1 - Matriz de indicadores de desempenho e classificação). As avaliações tiveram foco em valores extremos (valores mínimo e máximo), seus valores médios, as medidas separatrizes, como os quartis, e as medidas de dispersão, bem como o desvio padrão e o coeficiente de variação. Estas variáveis utilizadas são lembradas a seguir:

- CP: coeficiente de compacidade ou compacidade;
- CR: coeficiente de circulação ou circulação;
- DS: coeficiente de densidade de alvenarias ou densidade de alvenarias;
- FC: coeficiente de fachada ou fachada;
- OC: número de ocorrências críticas;
- OM: número de ocorrências moderadas;
- OL: número de ocorrências leves;
- OPC: número de ocorrências ponderadas críticas;

[1] Referências sobre estatística: (PINHEIRO *et al.*, 2009).

- OPM: número de ocorrências ponderadas moderadas;
- OPL: número de ocorrências ponderadas leves.

A fim de se verificar o comportamento das variáveis, foi realizada a análise descritiva, conforme os resutados da Tabela 3:

Tabela 3 - Medidas descritivas da amostragem ciclo AVALIAR

Categorias	Variável	Min	Max	Q1	Q2	Q3	Média	DP	CV (%)
Coeficientes	CP	0,28	0,46	0,32	0,34	0,37	0,35	0,04	11,43
	DS	0,06	0,08	0,06	0,07	0,07	0,07	0,00	0,00
	CR	0,29	0,43	0,34	0,36	0,38	0,36	0,03	8,33
	FC	0,39	1,06	0,74	0,84	0,93	0,83	0,13	15,66
Ocorrências	OC	0,00	24,00	4,00	6,00	9,00	6,95	4,92	70,79
	OM	1,00	29,00	12,00	14,00	17,00	14,91	4,84	32,46
	OL	0,00	8,00	1,00	2,00	3,00	1,98	1,69	85,35
Ocorrências ponderadas	OPC	0,00	24,00	4,00	6,00	9,00	6,95	4,92	70,79
	OPM	1,00	15,00	6,00	7,00	9,00	7,68	2,43	31,64
	OPL	0,00	2,00	0,00	1,00	1,00	0,59	0,57	96,61

Min - Valor mínimo; Max - Valor máximo; DP - Desvio-padrão; CV (%) - Coeficiente de variação em forma de porcentagem.

Fonte: A autora (2020).

A partir da Tabela 3, foram obtidos os seguintes resultados:

- Compacidade (CP) possui média de 0,35 e um coeficiente de variação de 11,43%, sendo considerado um valor moderado, podendo afirmar que a variabilidade dessa variável é baixa e os dados estão moderadamente concentrados em torno da média;
- Densidade de alvenarias (DS) possui média 0,07 e uma variabilidade extremamente baixa, indicada pela proximidade dos valores extremos e dos quartis, o que indica que os dados estão bastante concentrados em torno da média;

- Circulação (CR) possui um valor médio de 0,36 e um coeficiente de variação de 8,33%, indicando uma baixa variabilidade dos dados, ou seja, os dados estão bem próximos do valor médio;
- Fachada (FC) possui o valor médio de 0,83 e uma variabilidade também considerada moderada, devido ao seu coeficiente de variação de 15,66%, indicando que os dados estão moderadamente concentrados em torno da média;
- O número médio de ocorrências críticas (OC) é aproximadamente 7, porém sua variabilidade é bastante alta, com um coeficiente de variação de 70,79%, os dados estão bastante dispersos com relação ao seu valor médio;
- Ocorrências moderadas (OM) possuem média de aproximadamente 15, também com uma variabilidade considerada alta, com um coeficiente de variação de 32,46%, os dados não estão bem concentrados em torno da média;
- O número médio de Ocorrências leves (OL) é de aproximadamente 2, também com uma variabilidade bastante alta, com um coeficiente de variação de 85,35%, os dados estão bastante dispersos com relação à média;
- Para as ocorrências ponderadas, tem-se que críticas (OPC) e moderadas (OPM) possuem valores médios bastante próximos, sendo 6,95 e 7,68, respectivamente, e um valor médio de 0,59 para ocorrências ponderadas leves (OPL). Todas as ocorrências ponderadas mantiveram a alta variabilidade, com valores muito altos para o coeficiente de variação.

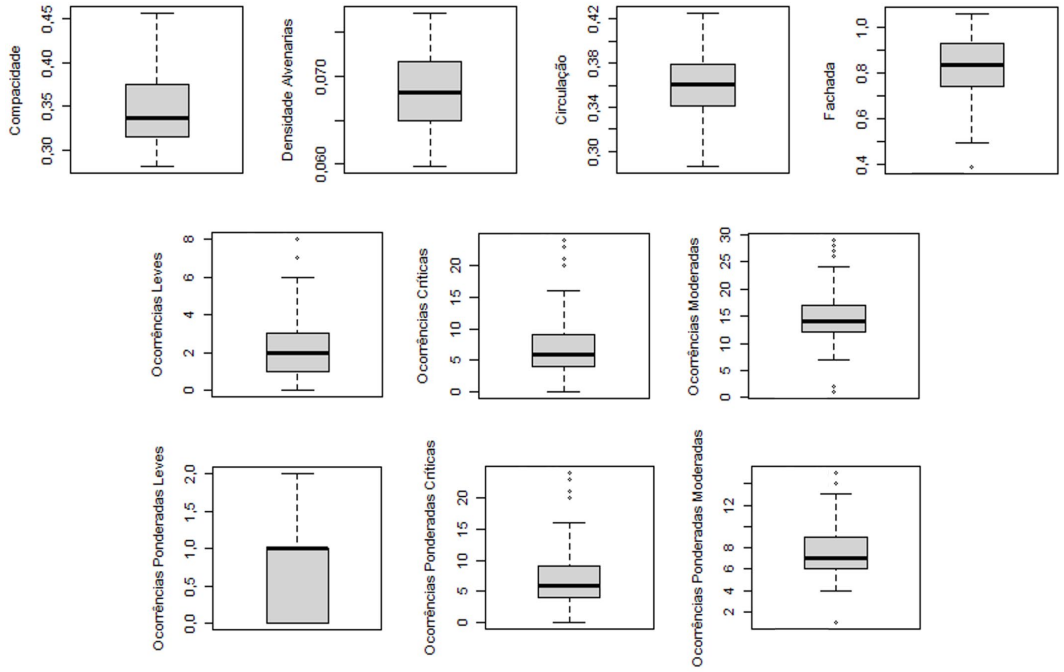
Na Figura 60 verificou-se a representação do comportamento das variáveis em *boxplots*[1], de onde foram extraídas as seguintes análises:

- Compacidade (CP), Densidade de alvenarias (DS), Circulação (CR) e Ocorrências ponderadas leves (OPL) não possuem *outliers*;
- A variável Fachada (FC) possui um *outlier*, o indivíduo TAG 56;
- Ocorrências críticas (OC) apresentam os indivíduos TAG 40, 56, 82 e 90 como *outliers*;

[1] *Boxplot* ou gráfico de caixa é um tipo de gráfico utilizado para auxiliar na identificação de valores discrepantes de um conjunto de dados (*outliers*), assim como comparar a dispersão entre dois ou mais conjuntos de dados (análise da simetria). A altura do retângulo é definida pelos quartis Q1 e Q3. Uma linha secciona o retângulo no valor da mediana (ou Q2). A haste central se estende do valor mínimo ao máximo do conjunto de dados (MORETTIN; BUSSAB, 2017).

- Ocorrências moderadas (OM) apresentam os indivíduos TAG 19, 36, 71, 83, 93 e 96 como *outliers*;
- Ocorrências leves (OL) apresentam os indivíduos TAG 12 e 16 como *outliers*;
- Ocorrências ponderadas críticas (OPC) apresentam os indivíduos TAG 40, 56, 82 e 89 como *outliers*;
- Ocorrências ponderadas moderadas (OPM) apresentam os indivíduos TAG 19, 36, 71, 83 e 93 como *outliers*.

Figura 60 - *Boxplots* para variáveis do ciclo AVALIAR



Fonte: A autora (2020).

Enfim, para selecionar os indivíduos da amostragem mais aptos a seguirem para o ciclo "Evoluir", foram classificados os 30 indivíduos[1] que tiveram melhor desempenho. Para esta classificação, foi calculado o desvio absoluto entre cada variável em relação a seus respectivos valores de referência. Em seguida, realizou-se, em cada variável, uma dicotomização (divisão em duas categorias), onde os 30 menores desvios absolutos receberam o valor 1 e os demais receberam valor 0. Na sequência, foi calculado um escore para cada indivíduo (TAG), realizando a soma dos seus respectivos valores dicotômicos em cada variável. Por exemplo, se determinado indivíduo obtiver um dos 30 menores desvios absolutos em 6

[1] Número arbitrado.

variáveis, seu escore será equivalente a 6. Os resultados da classificação estão contidos na Tabela 5.

Dado o empate de escores verificado na Tabela 4, foram então classificados os 36 indivíduos que apresentaram melhor desempenho.

Tabela 4 - Classificação dos indivíduos para o ciclo EVOLUIR

Indivíduo	Escore	Indivíduo	Escore	Indivíduo	Escore	Indivíduo	Escore
TAG 17	**8**	**TAG 21**	**4**	TAG 47	3	TAG 77	2
TAG 03	**6**	**TAG 24**	**4**	TAG 49	3	TAG 82	2
TAG 07	**6**	**TAG 26**	**4**	TAG 64	3	TAG 86	2
TAG 08	**6**	**TAG 40**	**4**	TAG 71	3	TAG 92	2
TAG 16	**6**	**TAG 42**	**4**	TAG 73	3	TAG 94	2
TAG 23	**6**	**TAG 48**	**4**	TAG 74	3	TAG 96	2
TAG 25	**6**	**TAG 54**	**4**	TAG 84	3	TAG 33	1
TAG 27	**6**	**TAG 61**	**4**	TAG 87	3	TAG 34	1
TAG 02	**5**	**TAG 63**	**4**	TAG 90	3	TAG 35	1
TAG 10	**5**	**TAG 66**	**4**	TAG 97	3	TAG 55	1
TAG 11	**5**	**TAG 93**	**4**	TAG 14	2	TAG 69	1
TAG 12	**5**	**TAG 98**	**4**	TAG 20	2	TAG 76	1
TAG 18	**5**	TAG 04	3	TAG 31	2	TAG 78	1
TAG 37	**5**	TAG 06	3	TAG 36	2	TAG 79	1
TAG 53	**5**	TAG 22	3	TAG 44	2	TAG 85	1
TAG 65	**5**	TAG 28	3	TAG 51	2	TAG 88	1
TAG 70	**5**	TAG 29	3	TAG 56	2	TAG 89	1
TAG 83	**5**	TAG 30	3	TAG 57	2	TAG 91	1
TAG 0	**4**	TAG 32	3	TAG 58	2	TAG 99	1
TAG 01	**4**	TAG 38	3	TAG 59	2	TAG 50	0
TAG 05	**4**	TAG 39	3	TAG 62	2	TAG 52	0
TAG 09	**4**	TAG 41	3	TAG 67	2	TAG 60	0
TAG 13	**4**	TAG 43	3	TAG 68	2	TAG 80	0
TAG 15	**4**	TAG 45	3	TAG 72	2	TAG 81	0
TAG 19	**4**	TAG 46	3	TAG 75	2	TAG 95	0

Fonte: A autora (2020).

Com a mesma classificação é possível verificar que o indivíduo TAG17 obteve o melhor desempenho, obtendo um dos 30 menores desvios em 8, de um total de 10 variáveis. Observa-se também que os indivíduos TAG50, 52, 60, 80, 81 e 95 obtiveram os piores desempenhos, não obtendo um dos 30 menores desvios em nenhuma variável.

- Resultados do ciclo “Evoluir”

Embora os critérios de otimização do AG tenham sido definidos apenas para os indicadores Compacidade (CP) e Circulação (CR), optou-se por coletar os dados de todos os indicadores novamente, para que fosse possível compreender não somente os resultados dos coeficientes otimizados, como também um possível impacto nos demais indicadores. Assim, após a execução do AG, seguiu-se o mesmo critério de análises descritivas do clico anterior e os resutados são apresentados na Tabela 5:

Tabela 5 - Medidas descritivas dos indivíduos ciclo EVOLUIR

Categorias	Variável	Min	Max	Q1	Q2	Q3	Média	DP	CV (%)
Coeficientes	CP	0,464	0,508	0,468	0,480	0,494	0,483	0,015	3,154%
	DS	0,066	0,081	0,073	0,075	0,076	0,075	0,003	3,796%
	CR	0,275	0,362	0,301	0,309	0,336	0,316	0,025	7,776%
	FC	0,518	0,675	0,582	0,589	0,595	0,592	0,028	4,787%
Ocorrências	OC	0,00	5,00	0,00	2,00	2,00	1,405	1,481	105,351%
	OM	9,00	23,00	14,00	17,00	19,00	16,05	3,358	20,916%
	OL	0,00	6,00	0,00	0,00	4,00	2,054	2,285	111,222%
Ocorrências ponderadas	OPC	0,00	5,00	0,00	2,00	2,00	1,405	1,481	105,351%
	OPM	4,50	11,50	7,00	8,50	9,50	8,027	1,679	20,916%
	OPL	0,00	1,50	0,00	0,00	1,00	0,514	0,571	111,222%

Min - Valor mínimo; Max - Valor máximo; DP - Desvio-padrão; CV (%) - Coeficiente de variação em forma de porcentagem.

Fonte: A autora (2020).

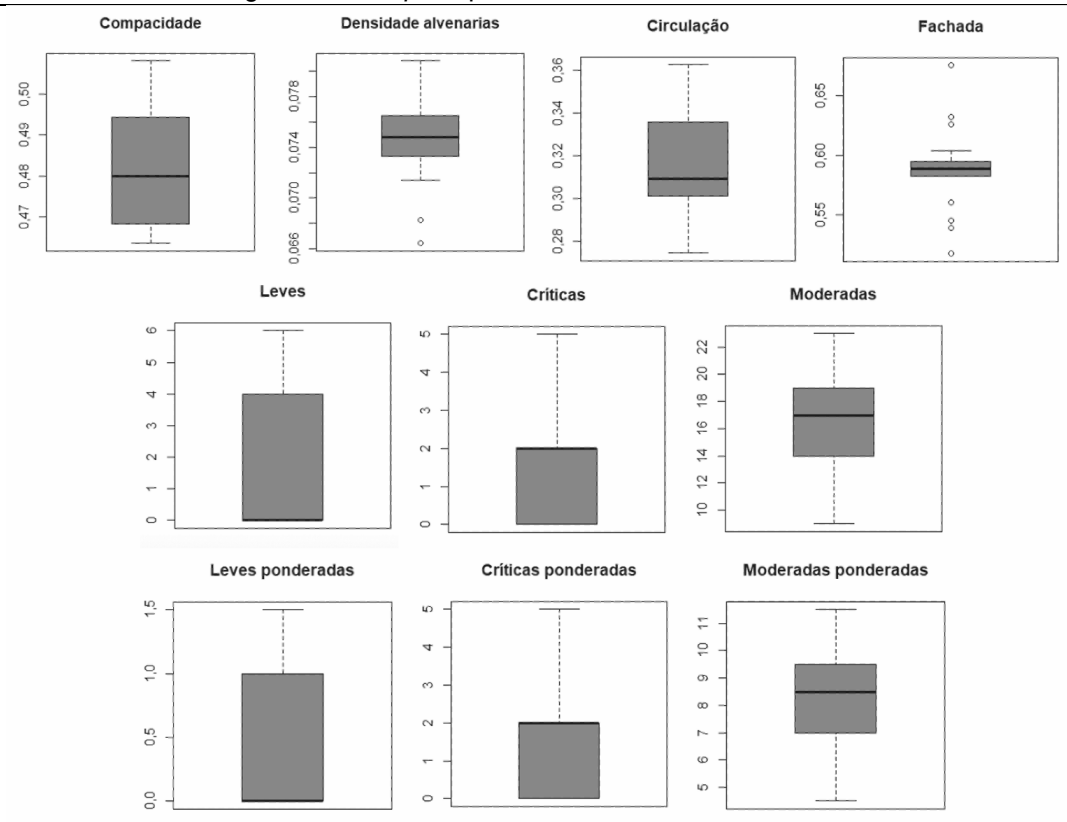
A partir da Tabela 5, foram obtidos os seguintes resultados:

- Compacidade (CP) possui média de 0,483 e um coeficiente de variação de 3,154%, indicando uma baixa variabilidade dos dados, ou seja, os dados estão bem próximos do valor médio;
- Densidade de alvenarias (DS) apresentou valor mínimo de 0,066 e máximo de 0,081, com valor médio foi de 0,075 e um coeficiente de variação de 3,796%, indicando uma baixa variabilidade dos dados, ficando próximos ao valor médio;
- Circulação (CR) possui um valor médio de 0,316 e um coeficiente de variação de 7,776%, indicando uma baixa variabilidade dos dados, ou seja, os dados estão bem próximos do valor médio;
- Fachada (FC) possui o valor médio de 0,592 e uma variabilidade também considerada baixa, devido ao seu coeficiente de variação de 4,787%, indicando que os dados estão moderadamente concentrados em torno da média;
- O número médio de ocorrências críticas (OC) é aproximadamente 1,405 e coeficiente de variação muito alto, o que indica dispersão nos dados;
- Ocorrências moderadas (OM) possuem média de aproximadamente 16,054 sendo uma moderada dispersão de dados;
- O número médio de Ocorrências leves (OL) é de 1,481, sendo que o valor mínimo que a mesma recebeu foi de 0 e o máximo de 6;
- Para as ocorrências ponderadas, tem-se que críticas (OPC), moderadas (OPM) e leves (OPL) possuem valores médios de 1,405, 8,027 e 5,014, respectivamente. As ocorrências OPC e OPL tiveram alta variabilidade de dados e a ocorrência OPM apresentou variabilidade média.

Na Figura 61 verificou-se a representação do comportamento das variáveis em *boxplots,* de onde foram extraídas as seguintes análises:

- As variáveis Compacidade (CP), Circulação (CR) e todas as categorias de ocorrências não possuem *outliers*;
- Na variável Fachada (FC) os indivíduos TAG 15, 16, 27, 35, 18, 21, 23, 25, 26, 28 e 31 são *outliers*.

Figura 61 - *Boxplots* para variáveis do ciclo EVOLUIR



Fonte: A autora (2020).

- Comparação de resultados dos ciclos “Avaliar” e “Evoluir”

Calculados os ciclos “Avaliar” e “Evoluir”, foram então realizadas as comparações entre os seus resultados para confirmação da hipótese de otimização dos indicadores Compacidade (CP) e Circulação (CR), quando da aplicação do AG. Para efeitos de observação de interferências entre as variáveis, foram comparados também os resultados dos demais indicadores.

As variáveis foram comparadas de acordo com seus valores de referência e a Tabela 6 apresenta estes comparativos:

Tabela 6 - Comparação das variáveis entre ciclos AVALIAR e EVOLUIR

Categoria	Variável	Média ciclo Avaliar	Média ciclo Evoluir	Valor de referência
Coeficientes	**CP**	0,350	0,483	Ideal = 0,886
	DS	0,070	0,075	Máx. = 0,160
	CR	0,360	0,316	Máx. = 0,250
	FC	0,830	0,592	Máx. = 0,650
Ocorrências	**OC**	6,950	1,405	Ideal = 0,000
	OM	14,910	16,054	Ideal = 0,000
	OL	1,980	2,054	Ideal = 0,000
Ocorrências ponderadas	**OPC**	6,950	1,405	Ideal = 0,000
	OPM	7,680	8,027	Ideal = 0,000
	OPL	0,590	0,514	Ideal = 0,000

Fonte: A autora (2020).

- O indicador Compacidade (CP) apresentou diferença significativa entre os ciclos, sendo que os dados do ciclo “Evoluir” apresentaram maior aproximação ao valor de referência, comprovando a sua otimização;
- O indicador Circulação (CR) apresentou redução entre os ciclos, sendo que os dados do ciclo “Evoluir” apresentaram maior aproximação ao valor de máximo de referência, comprovando a sua otimização;
- Densidade das alvenaria (DS) apresentou sutil perda, entretanto, manteve-se com ótimo resultado, abaixo do máximo de referência;
- Fachada (FC) demonstrou ser um indicador impactado positiva e significativamente pela otimização, passando a respeitar o limite máximo desejado.

As Ocorrências e Ocorrências Ponderadas oscilaram entre aumento e diminuição dos seus valores médios. Entretanto, observando-se as tabelas 1 e 2, nota-se que houve otimização favorável no número total das Ocorrências Ponderadas, conforme compara a Tabela 7.

Tabela 7 - Ocorrências ponderadas totais entre ciclos AVALIAR e EVOLUIR

Ocorrências ponderadas totais		ciclo Avaliar	ciclo Evoluir
Total de ocorrências	**máximo**	28	14
	mínimo	1	7
	média	14,98	9,95

Fonte: A autora (2020).

O total de Ocorrências Ponderadas apresentou diferença entre os ciclos, sendo que os dados do ciclo “Evoluir” apresentaram significativa diminuição do número médio destas ocorrências. Os valores da Tabela 7 demonstram assim, impacto favorável da otimização mesmo em indicadores não tratados diretamente.

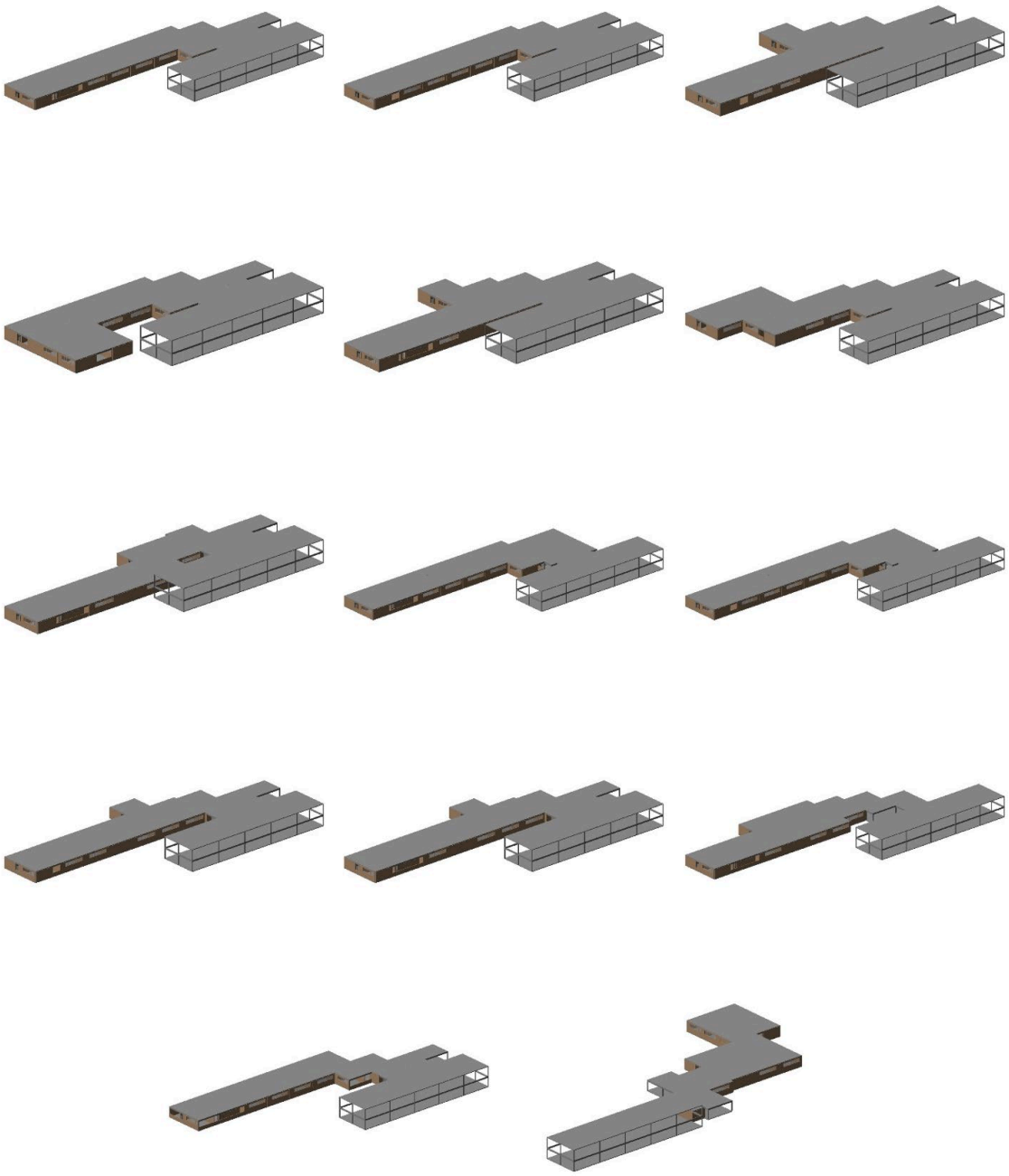
Diante aos indicadores analisados no ciclo “Evoluir”, conforme o mesmo critério do ciclo “Avaliar”, foram listados os indivíduos que obtiveram o melhor escore relativo final. Tais indivíduos, em modelos BIM IFC são apresentados na galeria da Figura 62.

Tabela 8 - Classificação dos novos indivíduos após o ciclo EVOLUIR

Indivíduo	Escore	Indivíduo	Escore
TAG 01	10	**TAG 10**	10
TAG 00	10	**TAG 11**	10
TAG 02	10	**TAG 12**	10
TAG 03	10	**TAG 13**	10
TAG 04	10	**TAG 15**	10
TAG 05	10	**TAG 17**	10
TAG 09	10	**TAG 19**	10

Fonte: A autora (2020).

Figura 62 - Indivíduos otimizados no ciclo EVOLUIR



Fonte: A autora (2020).

5.2 Discussão

No capítulo “Método da Pesquisa”, na seção 3.4, que delineia os critérios de Avaliação do constructo, foram planejadas quatro questões de discussão sobre as contribuições prática e teórica do trabalho:

1) a inserção de um *script* de algoritmo genético evolucionário no processo de *design* generativo é exequível?
2) a integração das estratégias de geração, avaliação e evolução de uma solução de projeto foi viabilizada computacionalmente?
3) houve incremento nos indicadores de desempenho das soluções produzidas generativamente após a implementação do algoritmo evolucionário?
4) verifica-se a asserção do termo BIM Evolucionário (E-BIM) no contexto dos algoritmos de inteligência artificial aplicados a projetos de engenharia e arquitetura? Quais indícios sustentariam essa premissa?

Com base nos resultados quantitativos da seção 5.1, os números são notórios a respeito da capacidade generativa de soluções válidas para o objeto prova de conceito do “Ciclo Gerar”. É nítido como o estabelecimento de uma gramática da forma para um problema definido, aliado ao processamento computacional de um algoritmo generativo, permite ao autor do projeto uma exploração formal muito mais ampla se comparada a um método de projeto não generativo. O método proposto por esta pesquisa ainda vai além: coloca à disposição das partes interessadas não somente um espectro de variedade formal em suas formas primárias: entrega alternativas em modelos BIM IFC que proporcionam explorações e análises muito mais aprofundadas, dada sua característica paramétrica de dados.

Ocorre que para muitos projetistas, como coloca Nourian (2016), a avaliação qualitativa do projeto requer informações contextuais que não são necessariamente codificadas pois existem qualidades que não são facilmente representáveis ou mensuráveis. Para o autor, sempre haverá necessidade de visão especializada e envolvimento de partes interessadas reais, e, portanto, a questão da avaliação é tratada por meio de discussões.

No entanto, Silva (2005) reflete a partir da ótica de que questões pertinentes à avaliação da qualidade do projeto arquitetônico se consolidam, em última análise, na emissão de um juízo de valor daquele que avalia. Sendo, para o autor, a natureza desse juízo na arquitetura uma questão de opinião. Portanto, mesmo uma avaliação humana sobre a qualidade de um projeto arquitetônico não será definitiva, visto que também está condicionada à opinião que se tem sobre o avaliador.

Para Guimarães *et al.* (2020), a crescente utilização de sistemas de IA tem levado a um impacto significativo na sociedade e nas suas decisões cotidianas. No entanto, os humanos tendem a ser menos dispostos a aceitar decisões feitas pelas máquinas quando não são diretamente interpretáveis ou confiáveis. A interpretabilidade pode ser definida pela capacidade de explicar ou apresentar resultados em termos compreensíveis a um ser humano.

Neste sentido, um processo de avaliação automatizado, baseado em informações paramétricas, requer uma síntese de muitas análises e uma estrutura sobre a qual as medidas de desempenho possam ser julgadas quanto à sua qualidade relativa. Isso pode ser feito de acordo com indicadores de desempenho previamente definidos para o problema (NOURIAN, 2016).

Os resultados quantitativos de avaliação automatizada do “Ciclo Avaliar” são objetivos em relação aos indicadores previamente fundamentados e então coletados. Resultados estes que demonstram recursividades analíticas na abordagem interpretativa e, portanto, transmitem confiabilidade sobre o processo avaliativo adotado. Este ciclo demonstrou cumprir com efetividade a classificação dos melhores candidatos ao “Ciclo Evoluir”.

Newell e Simon (1972) defendem que qualquer decisão racional pode ser vista como uma conclusão alcançada a partir de certas premissas, e que o comportamento de uma pessoa racional pode ser verificado, portanto, se o valor e

as premissas factuais sobre as quais ela baseia suas decisões forem especificados para ela (*apud* BUCHANAN, 2005).

A automação está movendo a realidade do *design* computacional para além das narrativas BIM e *design* paramétrico, que dominaram a arquitetura e a engenharia nas últimas duas décadas. Algoritmos de IA fornecem ao projetista um *feedback* sobre as decisões de projeto. Não apenas a validação de uma ideia, mas uma sugestão de uma projetação em curso (ANDIA; SPIEGELHALTER 2014).

E foi justamente o que se experimentou no “Ciclo Evoluir”. O algoritmo evolucionário genético, inserido através de um *script* no fluxo do algoritmo, produziu indivíduos melhorados, com base na combinação dos melhores existentes na população, e, através da estratégia da sobrevivência do mais apto, eliminou os menos aptos.

Algoritmos evolucionários aplicados ao *design* não tentam encontrar a solução ótima global, mas encontram uma solução que satisfaça alguns critérios enquanto sacrifica outros. Otimizam as soluções existentes selecionadas, e, então, as soluções melhoradas se propagam por toda a população. Embora seja capaz para o algoritmo encontrar o ótimo global, tal resultado não pode ser garantido.

Torna-se fundamental, no entanto, refletir que a mesma incerteza sobre encontrar a “solução ótima” ocorre durante o processo de criação produzido puramente por um humano. Ou seja, é inviável, mesmo para um projetista humano experiente, maximizar todos os objetivos simultaneamente. Então, em vez disso, procura-se encontrar soluções adequadas para o maior número de objetivos possíveis simultaneamente. Projetistas humanos nem sempre possuem as informações completas a respeito de um problema, não têm tempo ilimitado para chegar a uma conclusão, e são naturalmente limitados pela capacidade cognitiva.

Neste sentido, deve-se enfatizar que a proposta deste método e as ferramentas não se propõem a substituir os projetistas ou sua tomada de decisão. Esses instrumentos devem ser usados como recursos interativos e serão tão bons quanto

forem aqueles que programarem os *scripts* e as informações geradas em colaboração com as partes envolvidas na criação do projeto.

Um *script* aberto incorporado ao processo apresenta a vantagem de não ser uma “caixa-preta”, como ocorre quando um *plug-in* é fixado. Na seção 4.3, o Galapagos, aplicativo evolucionário nativo do Grasshoperr, foi colocado como exemplo de um processo fechado que permite ajustes limitados na programação de busca do algoritmo. No processo generativo-evolucionário E-BIM ora proposto, o *script* do algoritmo genético é adaptável e permite ajustes conforme se desenvolve a interação projetista/algoritmo. As prioridades para os critérios de otimização podem variar conforme o projeto: performance espacial, desempenho ambiental, economicidade, construtibilidade, dentre outras tantas possibilidades.

Olhando para a contínua evolução das pesquisas e métodos, é oportuno lembrar a abordagem de Pratschke (2015) quando o autor reflete sobre a previsão do ciberneticista Heinz von Foester, que previu, já nos anos 1960, que ao invés de focar no projeto de um objeto mecânico, há que propor um sistema orgânico onde haja a mudança de foco na cibernética do mecanismo para linguagem e de sistemas observados para sistemas que observam. Em outras palavras, como postula a Cibernética de Segunda Ordem, um sistema inteligente que permita a criação conjunta de objetivos e que assim o processo permita também a inclusão da subjetividade, em algum grau.

Os resultados quantitativos que comparam os indicadores da implementação do constructo comprovam que a combinação de fluxos paramétricos com soluções evolutivas resultou em uma modelagem BIM já otimizada. O uso desta abordagem aumenta a capacidade de tomada de decisão do projetista, pois o assiste na geração de múltiplas alternativas, na avaliação para comparação das opções e na otimização computacional daquelas mais potenciais a seus objetivos estabelecidos. Dispor desta inteligência adicional pode aumentar potencialmente a capacidade de tomada de decisão da equipe do projeto. O recurso de geração de alternativas e simulações é particularmente útil em cenários de projeto complexos.

Muitas vezes, espera-se da inteligência artificial efeitos de ficção científica, como os citados na abertura do capítulo “Introdução” deste trabalho. Mas, em ciência, resultados são construídos paulatinamente. A IA viável, e aquela que vemos neste trabalho, é um incremento que aumenta nossas capacidades humanas como criadores, acentuando nossas capacidades e compensando nossos pontos mais fracos. A tecnologia demonstra não nos substituir, mas nos tornar mais eficientes em nossos processos.

Com uma mentalidade mais esclarecida a respeito da evolução, e à medida que aumentarem a disseminação e a adesão da aplicabilidade desses algoritmos de IA em contextos próprios, mais o refinamento da teoria proposta contribuirá para modificar o *status quo* das nossas carências como sociedade.

Em projeto, a fundamentação da decisão pode estar em dois momentos distintos: na justificativa para uma escolha ou na verificação de seu efeito. Ou seja, pode-se decidir por uma determinada alternativa considerando-se um argumento verdadeiro, e pode-se decidir por uma alternativa e avaliar suas consequências. Formular e verificar as hipóteses é a base do pensamento científico (MOREIRA, 2007).

Para Lacerda (2012), uma das razões para aplicação da metodologia DSR nas pesquisas é a possibilidade deste método diminuir a lacuna existente entre teoria e prática e tornar um constructo generalizável a uma determinada classe de problemas. Conforme discorre em sua narrativa, esta generalização permite que o conhecimento gerado em uma situação particular possa, posteriormente, ser aplicado a outras situações similares enfrentadas pelas mais diversas organizações.

Em consonância a este entendimento a respeito da DSR, a aplicabilidade do constructo E-BIM pode, portanto, tornar-se generalizável, na medida em que se estrutura em sistemas abertos e configuráveis, tais como a Gramática da Forma, o sistema generativo de VPL, *script* do AG em linguagem aberta e resultados em modelos paramétricos neutros (IFC).

Como visto no início deste capítulo, embora não totalmente automatizado nativamente, o fluxo construído demostrou-se viavelmente produtivo e eficiente para um contexto de produção de concepção de projetos, conforme visto no experimento.

No que tange à terminologia E-BIM ora proposta, ocorre que durante manifestação da problemática desta pesquisa, foi observada a oportunidade de proposição de um termo próprio que definisse a integração dos campos da computação evolucionária e dos projetos paramétricos, em especial os projetos BIM. Assim, foi proposto:

> BIM Evolucionário (E-BIM): método aplicado a projetos de edificações, no qual modelos virtuais paramétricos são gerados, avaliados e otimizados pela integração de algoritmos generativos e evolucionários. Explora a aplicabilidade dos algoritmos de inteligência artificial no fluxo de desenvolvimento e automação destes projetos.

A asserção da terminologia, no contexto dos algoritmos de inteligência artificial aplicados a projetos de edificações, foi verificada por meio das respostas de 41 respondentes das áreas de Construção Civil (34,1%) e Ciências da Computação (65,9%). A inquirição apresentou unanimidade a favor da pertinência de aplicação da terminologia: “Inicialmente [a aplicabilidade do termo BIM Evolucionário] parece adequada, merecendo atenção para melhor avaliação em estudo mais aprofundado.”. O Apêndice E apresenta o questionário aplicado.
Diante de tal aceitabilidade, reforça-se também a contribuição teórica da pesquisa, com a proposição de um termo que especifica esta abordagem de modelagem evolucionária interoperável e multidisciplinar.

6 CONCLUSÕES

Tecnologias e *softwares* relativos ao BIM, *design* generativo, verificação automatizada de projetos são amplamente aplicados no mercado AEC, todavia a integração no processo proposto e a agregação dos algoritmos evolucionários, desta pesquisa demonstram contribuir para o incremento da geração otimizada de soluções para projetos de edificações, em especial na fase do *Early Concept Design,* avançando na aplicabilidade da inteligência artificial no campo da construção civil.

O *script* aberto de um AG, incorporado ao processo generativo-evolucionário, permite ajustes conforme se desenvolve a interação projetista/algoritmo. As estratégias e os critérios de otimização do algoritmo podem variar conforme as necessidades do projeto, demonstrando também seu atributo de adaptabilidade.

O “Ciclo Gerar” favorece uma exploração formal ampla, uma vez que alia, em um algoritmo paramétrico, o fundamento generativo da Gramática da Forma e o processamento computacional, entregando alternativas de projetos em modelos BIM IFC. No “Ciclo Avaliar”, a partir de indicadores previamente estipulados, ocorre a avaliação automatizada das alternativas geradas no ciclo anterior, que então são classificadas pelos resultados quantitativos objetivos, gerando-se confiabilidade e segurança acerca do processo avaliativo adotado. No “Ciclo Evoluir”, há a implementação do algoritmo evolucionário genético que otimiza as soluções geradas e classificadas anteriormente, retornando uma modelagem BIM já otimizada pelo algoritmo de AI, ampliando, assim, a capacidade de tomada de decisão do projetista.

Sintetizando o constructo, verifica-se a indicação da asserção da terminologia BIM Evolucionário (E-BIM) proposta, dizendo respeito ao método aplicado a projetos de edificações, no qual modelos virtuais paramétricos são gerados, avaliados e otimizados pela integração de algoritmos generativos e evolucionários, explorando-se, assim, a aplicabilidade dos algoritmos de inteligência artificial no fluxo de desenvolvimento e automação destes projetos. Todavia, vale destacar que a

terminologia foi verificada em um universo qualificado de entrevistados, com limitação quantitativa, merecendo também novas avaliações qualitativas em revisões científicas paritárias.

Enfatiza-se também que, mesmo com os resultados e conclusões favoráveis verificadas no experimento, para que estes possam ser tomados como generalizáveis, acredita-se que trabalhos futuros possam explorar novos cenários e condições de aplicabilidade do constructo. E ainda, que a evolução do constructo para um API própria possa catalisar a automação e otimização do processo e, assim, potencializar maiores ganhos futuros.

Conquanto, considera-se que o início da aplicabilidade destes elementos combinados em ambiente da práxis projetual já seja amplamente viável, conforme a presente autora vem experimentando no exercício da disciplina.

REFERÊNCIAS BIBLIOGRÁFICAS

ANDIA, A.; SPIEGELHALTER, T. **Post-Parametric Automation in Design and Construction**. London: Artech House, 2014. 230 p.

ANDRADE E SILVA, Flávio Paulino de. **Verificação automática dos requisitos de projetos da Norma de Desempenho pela plataforma BIM Solibri Model Checker**. 2017. 161 p. Dissertação (Mestrado em 2017) – Universidade Federal de Minas Gerais, Belo Horizonte, 2017.

AREL, I., *et al.* Deep Machine Learning - A New Frontier in Artificial Intelligence Research. **IEEE Computational Intelligence Magazine**, v. 5, p. 13-18, 2010.

ASHLOCK, D. **Evolutionary Computation for Modeling and Optimization**. New York: Springer, 2006. 572p.

AUTODESK. **AutoLISP Developer's Guide.** AUTODESK. Autodesk Knowledge Network. 2012. Disponível em: http://docs.autodesk.com/ACDMAC/2013/ENU/ PDFs/acdmac_2013_autolisp_developers_guide.pdf. Acesso em: 13 nov. 2019.

BARBOSA, F. *et al*. Reinventing construction through a productivity revolution. **McKinsey Global Institute**, [*s. l.*], 2017.

BARRETO, G. H. A. S. **Generative Design for Building Information Modeling.** 2016. 79 p. Dissertação (Mestrado em Sistemas da Informação e Engenharia da Computação) - Instituto Superior Técnico de Lisboa, Lisboa, 2016.

BELL, H. *et al.* **Standardized Computable Rules**. Oslo: Standards Norway, dez. 2009.

BENTLEY, P. J. An Introduction to Evolutionary Design by Computers. *In*: **Blog ResearchGate**. 1999. Disponível em: https://www.researchgate.net/ publication/ 244418116_. Acesso em: 14 out. 2019.

BLANCO, J. L. *et al.* Artificial intelligence: Construction technology's next frontier. *In*: **McKinsey & Company.** [s.n.], abr. 2018. Disponível em: https://www.mckinsey.com/business-functions/operations/our-insights/artificial-intelligence-construction-technologys-next-frontier. Acesso em: 2 out. 2019.

BOCK, T. The future of construction automation: Technological disruption and the upcoming ubiquity of robotics. **Automation in Construction**, [*s. l.*], v. 59, p. 113–121, 2015. Available at: https://doi.org/https://doi.org/10.1016/j.autcon.2015.07.022 Acesso em: 5 fev. 2021.

BOCK, T.; LINNER, T. (org.). Advanced Construction and Building Technology. *In*: **Robot-Oriented Design: Design and Management Tools for the Deployment of Automation and Robotics in Construction**. Cambridge: Cambridge University Press, 2015. p. 1–17. Available at: https://doi.org/DOI: 10.1017/ CBO9781139924146.002. Acesso em: 5 fev. 2021.

BORENSTEIN, Y.; MORAGLIO, A. **Theory and principled methods for the design of metaheuristics**. Natural Computing Series. Berlin: Springer, 2014. 270 p.

BRÍGITTE, G.; RUSCHEL, R. Identification of applicable patterns to algorithmization in BIM to explore solutions in the design stage of Social Housing. **Anais**… Conference of the iberoamerican society of digital graphics, 22. p. 68-73. 2018. Disponível em: https://www.researchgate.net/ publication/328677313. Acesso em: 10 nov. 2019.

BUCHANAN, Bruce G. **A (very) brief history of artificial intelligence**. AI Magazine, v. 26, n. 4, p. 53-53, 2005.

BUENO, E. G. **101 Conceitos de arquitetura e urbanismo na era digital**. São Paulo: Probooks, 2016. 256 p.

BYRNE, Jonathan. **Approaches to evolutionary architectural design exploration using grammatical evolution**. 2012. Tese University College Dublin, Dublin, 2012.

CAETANO, Inês *et al.* Aplicação do design generativo nas tecnologias BIM. **Anais...** Congresso português de building information modelling, 1., 2016. Braga: Universidade do Minho.

CALIXTO, Victor; CELANI, Gabriela. A literature review for space planning optimization using an evolutionary algorithm approach: 1992-2014. **Anais...** Conference of the Iberoamerican society of digital graphics 19., 2015, Florianópolis. p. 662-671.

CAMPANA, G. A.; OPLUSTIL, C. P. Conceitos de automação na medicina laboratorial: revisão de literatura. **J Bras Patol Med Lab**, Rio de Janeiro, v. 47, n. 2, p. 119-127, 2011.

CAMPOS, F. M. de**. Estudo do uso de parametrização e simulações computacionais nas etapas iniciais do processo de projeto visando à otimização**. 2017. 191 p. Dissertação (Mestrado em Arquitetura, Tecnologia e Cidade) – Faculdade de Engenharia Civil, Arquitetura e Urbanismo, Universidade Estadual de Campinas, Campinas, 2017.

CASTRO-LACOUTURE, D. *et al.* Concrete paving productivity improvement using a multi-task autonomous robot. **Automation and Robotics in Construction - Proceedings of the 24th International Symposium on Automation and Robotics in Construction**, [*s. l.*], p. 223–228, 2007.

CAU/BR. Conselho de Arquitetura e Urbanismo do Brasil. **Tabelas de honorários de serviços de arquitetura e urbanismo do Brasil**. Módulo I: Remuneração do Projeto Arquitetônico de Edificações. Brasília, 2014.

CELANI, G. An educational experiment with shape grammars and computer applications. **International Journal of Design Computing**, Liverpool, v. 3, 2002.

CELANI, G. *et al.* A Gramática da Forma como Metodologia de Análise e Síntese em Arquitetura. **Conexão – Comunicação e Cultura**. Caxias do Sul, v. 5, n. 10, 2006.

CELANI, M. G. C.; VAZ, C. E. V. Scripts em CAD e ambientes de programação visual para modelagem paramétrica: uma comparação do ponto de vista pedagógico, 2011, **Anais...** V TIC, Salvador, p. 1-13.

CHANG, K. Multiobjective Optimization and Advanced Topics. In: CHANG, K. **Design Theory and Methods using CAD/CAE**. London: Academic Press, 2015. p. 325-406.

COSTA, M. A; THADEU, M.; FAVARÃO, C. B. (Org.) **A Nova agenda urbana e o Brasil**: insumos para sua construção e desafios a sua implementação. Brasília: IPEA, 2018. 133 p.

CRESWELL, J. W. **Projeto de pesquisa**: métodos qualitativo, quantitativo e misto. Trad. Magda Lopes. 3.ed. Porto Alegre: Artmed, 2010. 296 p.

CUADROS, J. F. *et al.* A hybrid GA-SQP multi-objective optimization methodology for carbon monoxide pollution minimization in fluid catalytic cracking process. In: **Computer Aided Chemical Engineering**. Elsevier, 2013. p. 763-768.

DAVIS, D. **Modelled on Software Engineering**: flexible parametric models in the practice of architecture. Tese (Doutorado em Filosofia) - School of Architecture and Design, College of Design and Social context, RMIT University, Malborne, 2013. 243p.

DAVIS, L. **Handbook of Genetic Algorithms**. New York: Van Nostrand Reinhould, 1991. 385 p.

DE SIQUEIRA, E. C. **Heurísticas computacionais aplicadas a um problema flowshop híbrido multiobjetivo**. Tese (Doutorado em Modelagem Matemática e Computacional) – Centro Federal de Educação Tecnológica de Minas Gerais, Belo Horizonte, 2019. 150 p.

DOMINGOS, P. O. **O algoritmo mestre**: como a busca pelo algoritmo de machine learning definitivo recriará nosso mundo. São Paulo: Novatec, 2017. 344 p.

DUARTE, J. P. *et al.* A general indirect representation for optimization of generative design systems by genetic algorithms: Application to a shape grammar-based design system. **Automation in Construction**, v. 35, 2013.

EASTMAN, C. *et al.* Automatic rule-based checking of building designs. **Automation in Construction**, Elsevier, v. 18, n. 8, p. 1011-1033, 2009a.

EASTMAN, C. Automated Assessment of Early Concept Designs. **Architectural Design**, v. 79, p. 52-57. 2009b.

EASTMAN *et al.* **Manual de BIM**: Um guia de modelagem da informação da construção para arquitetos, engenheiros, gerentes, construtores e incorporadores. Tradução: Cervantes G. Alves Filho *et al*. Porto Alegre: Bookman, 2014. 500 p.

EIBEN, A. E.; SMITTH, J. E. **Introduction to Evolutionary Computing.** 2. ed. Berlim: Springer-Verlag, 2008. 316 p.

FACELI, K. *et al.* **Inteligência artificial**: uma abordagem de aprendizado de máquina. Rio de Janeiro: LTC, 2011. 394 p.

FDE - Fundação para o Desenvolvimento da Educação. **Arquitetura escolar paulista**: estruturas pré-fabricadas. FERREIRA, A. F.; MELLO, G. (org.). São Paulo, FDE, 2016. Disponível em: http://www.fde.sp.gov.br/PagePublic/Interna.aspx?codigoMenu=255. Acesso em: 7 set. 2019

FDE - Fundação para o Desenvolvimento da Educação. **Catálogo Ambientes**: Especificações da edificação. São Paulo, SP. FDE, 2019a. Disponível em: https://produtostecnicos.fde.sp.gov.br/ Pages/CatalogosTecnicos/ Default.aspx. Acesso em: 7 set. 2019

FDE - Fundação para o Desenvolvimento da Educação. **Prêmios e Menções de Arquitetura**. São Paulo, SP. FDE, 2019b. Disponível em: http://www.fde.sp.gov.br/PagePublic/Interna.aspx? codigoMenu=260. Acesso em: 7 set. 2019

FERREIRA, B.; LEITÃO, A. Design generativo para building information modeling. *In*: **I Congresso Português de Building Information Modelling**, 1., 2016. Braga: Universidade do Minho.

FISCHER, T.; HERR, C. M. Teaching Generative Design. *In*: **International Generative Art Conference**, 4. 2001, Milão. Disponível em: https://www.researchgate.net/publication/30869860_Teaching_Generative_Design /citation/download. Acesso em: 7 out. 2019.

FURTADO SILVA, N.; MACIEL FURTADO SILVA, L. Uma estratégia de ensino para estimular o uso de múltiplas linguagens arquitetônicas em resposta a tecnologias digitais emergentes. **Paranoá**: cadernos de arquitetura e urbanismo, n. 26, p. 185-204, 28 maio 2020.

GARCÍA DE SOTO, B. *et al.* Productivity of digital fabrication in construction: Cost and time analysis of a robotically built wall. **Automation in Construction**, [*s. l.*], v. 92, p. 297–311, 2018. Available at: https://doi.org/https://doi.org/10.1016/ j.autcon.2018.04.004. Acesso em: 12 fev. 2021.

GIPS, J. **Shape grammars and their uses**. Artificial Perception, Shape Generation and Computer Aesthetics. Stuttgart: Birkhauser Verlag, 1975. 243 p.

GUIMARÃES, F. G. **Aprendizagem e busca local em algoritmos meméticos para projeto assistido por computador**. 2008. 206p. Tese (Doutorado em

Engenharia Elétrica). Escola de Engenharia, Universidade Federal de Minas Gerais, Belo Horizonte, 2008.

GUIMARÃES, F. *et al.* **Applying Genetic Programming to Improve Interpretability in Machine Learning Models.** 2020. Disponível em: https://www.researchgate.net/publication/341505097.Acesso em: 29 out. 2020.

HILLER, J.; LIPSON, H. Automatic Design and Manufacture of Soft Robots. **IEEE Transactions on Robotics**, Evanston, v. 28, n. 2, p. 457-466, 2012.

HODGES, A. **Alan Turing**: The Enigma. Princeton: Princeton University Press, 2014. 768 p.

JIN, C. **A sequential process monitoring approach using hidden Markov model for unobservable process drift**. 2015. Tese de Doutorado. University of Cincinnati.

JO, Jun H.; GERO, John S. Space layout planning using an evolutionary approach. **Artificial Intelligence in Engineering**, v. 12, n. 3, p. 149-162, 1998.

JONES, N. **Architecture as a complex adaptive system.** 2009. Disponível em: https://www.researchgate.net/publication/307513051. Acesso em: 02 nov. 2019.

KHABAZI, Z. Generative algorithms with grasshopper version 2.0. **AA on-line book**, 2012. Disponível em: http://morphogenesism.com/wp/generative-algorithms. Acesso em: 02 nov. 2019.

KING, R. *et al.* Confronting the Urban Housing Crisis in the Global South: Adequate, Secure, and Affordable Housing. **World Resources Institute Working Paper**, 2017. Disponível em: https://files.wri.org/s3fs-public/towards-more-equal-city-confronting-urban-housing-crisis-global-south.pdf. Acesso em: 27 out. 2019.

KNIGHT, T. W. **Transformations in design**: a formal approach to stylistic change and innovation in the visual arts. Cambridge: Cambridge University Press, 1994. 276 p.

KOLAREVIC, B. **Architecture in the digital age**: design and manufacturing. 1.ed. London: Taylor & Francis, 2005. 320 p.

KOLO, S. J.; RAHIMIAN, F. P.; GOULDING, J. S. Offsite manufacturing construction: A big opportunity for housing delivery in Nigeria. **Procedia Engineering**, [*s. l.*], v. 85, p. 319–327, 2014. Available at: https://doi.org/10.1016/j.proeng.2014.10.557

KOTSIANTIS, Sotiris. **Supervised Machine Learning**: A Review of Classification Techniques. Informatica (Ljubljana), v. 31, 2007. Disponível em: https://www.researchgate.net/publication/265544297_Supervised_Machine_Learning_A_Review_of_Classification_Techniques. Acesso: 12 nov. 2019.

LACERDA, D. P. *et al.* **Design science research**: método de pesquisa para a engenharia de produção. São Leopoldo: [s.n.], 2012

LEE, Y. *et al.* Validations for ensuring the interoperability of data exchange of a building information model. **Automation in Construction,** v 58, p. 176-195, 2015.

LI, Y. Deep reinforcement learning. 2018. *In:* Cornell University. Disponível em: https://arxiv.org/pdf/1810.06339.pdf. Acesso em: 10. out. 2019.

LIN, S. H.; GERBER, D. J. Evolutionary energy performance feedback for design: multidisciplinary design optimization and performance boundaries for design decision support, **Energy Build**, v. 84, p. 426-441, 2014.

LOPES, H. S; TAKAHASHI, R. H. C. (org.) **Computação evolucionária em problemas de engenharia.** Curitiba: Omnipax, 2011. 385 p.

LUDOVICO, S. S. A.; BRANDÃO, D. Q. Caracterização da identidade morfológica do espaço arquitetônico de uma habitação evolutiva. **Gestão e Tecnologia de Projetos**, São Carlos, v. 13, n. 1, p. 39-58, 2018.

LUKKA, K. The constructive research approach. **Case study research in logistics. Publications of the Turku School of Economics and Business Administration, Series** B, v. 1, n. 2003, p. 83-101, 2003.

MANZIONE, L. **Proposição de uma estrutura conceitual de gestão do processo de projeto colaborativo com o uso do BIM**. 2013. 325p. Tese (Doutorado em Engenharia) - Escola Politécnica, Universidade de São Paulo, São Paulo, 2013.

MASCARÓ, J. **O custo das decisões arquitetônicas**. 4. ed. Porto Alegre: Masquatro. 2006. 192 p.

MOREIRA, D. C. **Os princípios da síntese da forma e a análise de projetos arquitetônicos**. Campinas, SP: [s.n.], 2007.

MORETTIN, P. A.; BUSSAB, W. O. **Estatística básica**. São Paulo: Saraiva, 2017.

NAGY, D. The problem of learning. *In:* **Medium** – Get smarter about what matters to you. Nova York, 2017a. Disponível em: https://medium.com/generative-design/generative-design-introduction-64fb2db38e1. Acesso em: 22 set. 2019.

NAGY, D. Learning from Nature. *In:* **Medium** – Get smarter about what matters to you. Nova York, 2017b. Disponível em: https://medium.com/generative-design/learning-from-nature-fe5b7290e3de. Acesso em: 22 set. 2019.

NAGY, D. Using Python in Grasshopper. *In:* **Medium** – Get smarter about what matters to you. Nova York, 2017c. Disponível em: https://medium.com/generative-design/using-python-in-grasshopper-77bfca86e84b. Acesso em: 17 nov. 2019.

NAGY, D. Nature-based hybrid computational geometry system for optimizing the interior structure of aerospace components. *In*: **ACM SIGGRAPH 2017 Talks**

(SIGGRAPH '17). Association for Computing Machinery, New York, NY, USA, 2017d, Article 76, 1–2. DOI: https://doi.org/10.1145/3084363.3085088

NASCIMENTO, M. F. P. do. **Arquitetura para a educação**: a contribuição do espaço para a formação do estudante**.** 2012. 154 p. Dissertação (Mestrado em História e Fundamentos da Arquitetura e do Urbanismo) – Faculdade de Arquitetura e Urbanismo de São Paulo, São Paulo, 2012.

NETO, S. P. Computação Evolutiva: desvendando os algoritmos genéticos. **Revista de Ubiquidade**, v. 1, n. 1, p. 34-45, 2011.

NETTO, V. M. O efeito da arquitetura: Impactos sociais, econômicos e ambientais de diferentes configurações de quarteirão. **Arquitextos**, São Paulo, ano 07, n. 079.07, Vitruvius, dez. 2006. Disponível em: https://www.vitruvius.com.br/ revistas/read/arquitextos/07.079/290. Acesso em: 02 nov. 2019.

NOURIAN, P. **Configraphics**: Graph theoretical methods for design and analysis of spatial configurations. Tese (Doutorado em Filosofia) - Delft University of Technology, Delft, 2016. 348p.

PEREIRA, P. R. P. *et al.* Analysis support for the design process of school 375 buildings. **Ambiente Construído**, Porto Alegre, v. 18, n. 3, p. 375-390, jul./set. 2018.

PINHEIRO, J. I.D *et al.* **Estatística básica**: a arte de trabalhar com dados. Rio de Janeiro: Elsevier Brasil, 2009.

PMI - Project Management Institute. **AI Innovators**: Cracking the Code on Project Performance. [S.l.], PMI, 2019. Disponível em: https://www.pmi.org/ learning/thought-leadership/pulse/ai-innovators. Acesso em: 22 set. 2019.

PRATSCHKE, A.; DI STASI, M.G. **Quão cibernética é a parametrização?** São Carlos, n. 11, 2015. Disponível em: http://www.nomads.usp.br/ virus/virus11/?sec=6&item=1&lang=pt. Acesso em: 15 set. 2020.

QUIRK, V. A Brief History of BIM. *In*: **ARCHDAILY**. 2012. Disponível em: https://www.archdaily.com/ 302490/a-brief-history-of-bim. Acesso em: 5 nov. 2019.

RMA - Robert Mcnell and Associates. **Grasshopper Wiki Pages**. 2016. Disponível em: https://wiki.mcneel.com/labs/explicithistory/home. Acesso em: 7 out. 2019.

ROLNIK, R. (Org). **Como fazer valer o direito das mulheres à moradia?** São Paulo: FAO/USP, 2012. Disponível em: <https://goo.gl/JmUjt5>. Acesso em: 5 fev. 2021.

RUSCHEL, R. C.; ANDRADE, M. L. V. X. de. Interoperabilidade de aplicativos BIM usados em arquitetura por meio do formato IFC. **Gestão & Tecnologia de Projetos**, São Paulo, v. 4, n. 2, nov. 2009.

RUSSELL, S.; NORVIG, P. **Inteligência Artificial**: uma abordagem moderna**.** Campus: São Paulo, Brazil, 2003. Disponível em https://books.google.com.br/books/about/Intelig%C3%AAncia_artificial.html?id=wBMvAAAACAAJ&redir_esc=y Acesso em 07 dez. 2019.

RUTTEN, David. Evolutionary Principles applied to Problem Solving. 2010. *In:* **Blog Grasshopper**. Algorithmic Modeling for Rhino, 2019. Disponível em: https://www.grasshopper3d.com/profiles/blogs/evolutionary-principles. Acesso em: 14 out. 2019.

SACKS, R.; BLOCH, Tanya. Comparing machine learning and rule-based inferencing for semantic enrichment of BIM models. **Automation in Construction**, v. 91, p. 256-272, 2018.

SACKS, R. *et al.* Parametric 3D modeling in building construction with examples from precast concrete. **Automation in Construction**, v. 13, n. 3, p. 291-312, 2004.

SAMMER, M. J. **The Expressive Power of Programming Languages in Architecture**. 2019. 108p. Tese (Master of Science Degree in Architecture) - Instituto Técnico de Lisboa, Lisboa, 2019.

SCHUMACHER, P. Parametricism: A New Global Style for Architecture and Urban Design. **Archit Design**, v. 79, p. 14-23, 2009.

SCHWABE, K; KÖNIG, M; TEIZER, J. BIM applications of rule-based checking in construction site layout planning tasks. *In*: **ISARC. Proceedings of the International Symposium on Automation and Robotics in Construction**. IAARC Publications, 2016. p. 1.

SEDREZ, M; MARTINO, J. A. Sistemas generativos**.** *In*: CELANI, M.G.C. *et al.* (org.). **Arquitetura contemporânea e automação**: prática e reflexão**.** São Paulo: ProBooks, 2018. 438 p.

SILVA, E. Crítica e avaliação no ensino do projeto arquitetônico: subsídios para uma discussão necessária, **Anais...** Seminário Projetar, 2005, Rio de Janeiro, UFRJ: Faculdade de Arquitetura e Urbanismo.

SILVA, F; ARANTES, E. Proposta de verificação automática dos requisitos de projeto pelo uso de ferramentas de análise BIM aplicados a norma de desempenho. *In*: **Encontro nacional de tecnologia do ambiente construído**, 16., 2016, São Paulo.

SILVA JÚNIOR, F. A. da. **O uso de algoritmos e de sistemas paramétricos na concepção arquitetônica de pequenas residências**. 2011. 147p. Dissertação (Mestrado em Arquitetura e Urbanismo) - Faculdade de Arquitetura e Urbanismo, Universidade de Brasília, Brasília, 2011.

SIMON, H. A. **The Sciences of the Artificial**. 3. ed. Cambridge: MIT Press, 1996. 248 p.

SMITH, D. K.; TARDIF, M. **Building Information Modeling**: A Strategic Implementation Guide for Architects, Engineers, Constructors, and Real Estate Asset Managers. Hoboken (NJ): John Wiley & Sons, 2009.

SOLIBRI, Inc. **Getting Started with Solibri Model Checker.** [S.l.]: SOLIBRI, 2016. Disponível em: https://solibri-assets.s3.amazonaws.com/old-site/2016/03/ Getting-Started-v9.pdf. Acesso em: 22 set. 2019

STINY, G. Introduction to shape and shape grammars. **Environment and planning B: planning and design**, v. 7, n. 3, p. 343-351, 1980.

STOVER, C.; WEISSTEIN, E. W. Parametric Equations. *In*: **MathWorld** - A Wolfram Web Resource. Disponível em: http://mathworld.wolfram.com/ ParametricEquations.html. Acesso em: 10 out. 2019

SUNG, W. **Grasshopper Learning Material**. 2010. Disponível em: https://woojsung.com/2010/04/09/digital-workshops-lectures-syracuse-architecture/. Acesso em: 12 out 2019.

TABARI, M. H. S. **Basic Genetic Algorithm in Grasshopper Python**. Nova York, 2020. Disponível em: https://medium.com/@saleh.tabari93/basic-genetic-algorithm-in-grasshopper-python-61fe1998f378. Acesso em: 05 jul. 2020.

TULLIO, F. B. M. (Org.) **Força, crescimento e qualidade da engenharia civil no Brasil**. Ponta Grossa: Atena, 2020. DOI: 10.22533/at.ed.873202109

TURING, Alan M. Computing machinery and intelligence. *In*: **Parsing the turing test.** Springer, Dordrecht, 2009. p. 23-65. DOI: https://doi.org/10.1007/978-1-4020-6710-5_3

VAISHNAVI, V.; KUECHLER, B. **Design Science Research in Information Systems**. Association for Information Systems. 2004. Disponível em: https://www.researchgate.net/publication/235720414_Design_Science_Research_ in_Information_Systems/. Acesso em: 25 set. 2019.

VASCONCELOS, J. A. de. **Computação evolucionária**: Otimização Evolutiva Multiobjetivo. Notas de aula, Departamento de Engenharia Elétrica, Universidade Federal de Minas Gerais, Belo Horizonte, 2016.

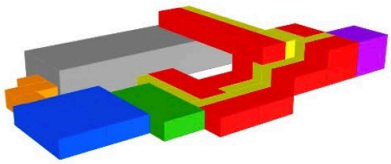
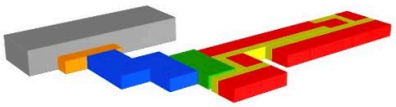
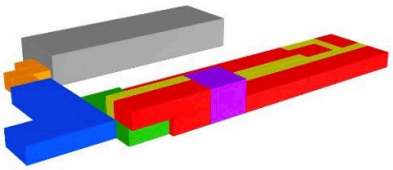
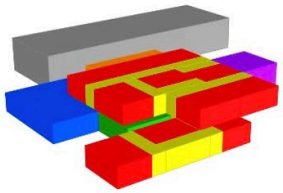
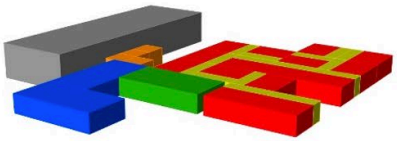
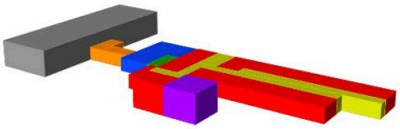
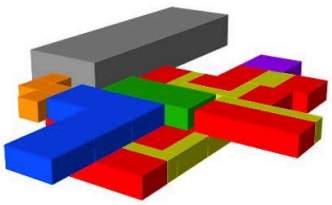
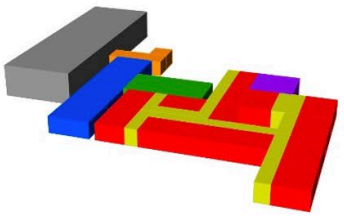
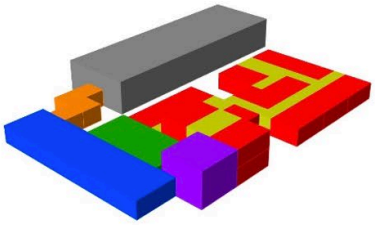
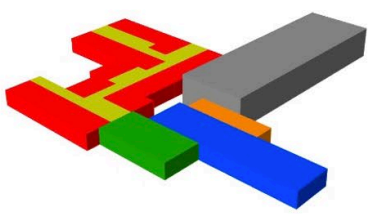
WORTMANN, T.; STOUFFS, R. Algorithmic complexity of shape grammar implementation. **Artificial Intelligence for Engineering Design, Analysis and Manufacturing**, Cambridge, v.32(2), p. 138-146, 2018.

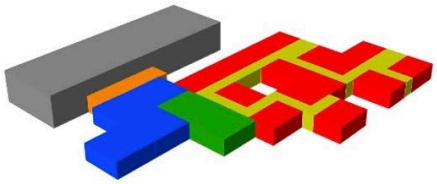
APÊNDICE A – Especificações do equipamento e *softwares* do experimento

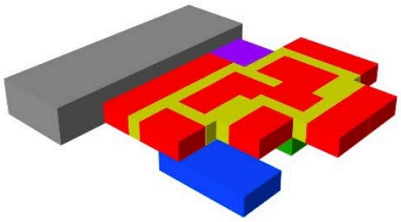
- Computador:
  - Nome do Modelo: iMac15,1
  - Monitor:
    - Tipo do Monitor: LCD Retina Integrado
    - Resolução: 5120 x 2880 Retina
  - Nome do Processador: Quad-Core Intel Core i5
  - Velocidade do Processador: 3,3 GHz
  - Número de Processadores: 1
  - Número Total de Núcleos: 4
  - Memória: 24 GB
  - Unidade física:
    - Capacidade: 880 GB (879.999.975.424 bytes)
    - Tipo de Mídia: SSD
    - Protocolo: SATA
  - BOOTCAMP:
    - Capacidade: 119,99 GB (119.993.794.560 bytes)

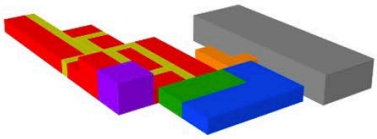
- *Softwares*:
  - Sistema operacional: Windows 10, Enterprise
  - *Design* generativo: Rhinoceros 6
    - *Plug-ins:* Grasshopper (nativo)
      Galapagos
      Grasshopper-Archicad Live Connection
      Tool Box
      Anemone
      Phyton (nativo)
  - Modelagem BIM: Archicad 22 Solo
  - Avaliação automatizada BIM: Solibri Model Checker
  - Tratamento de dados: Microsoft Excel

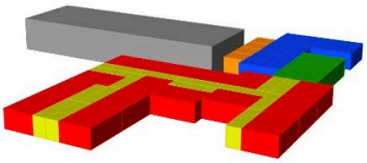
APÊNDICE B – Amostragem de indivíduos da população

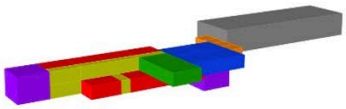











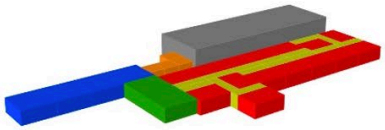


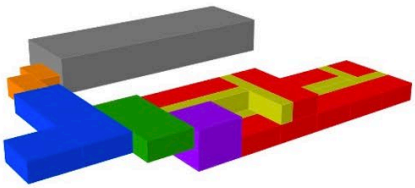


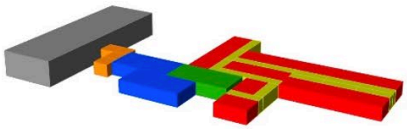


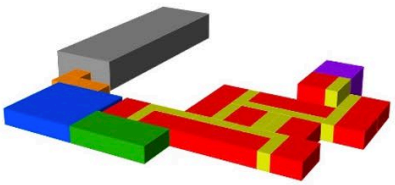


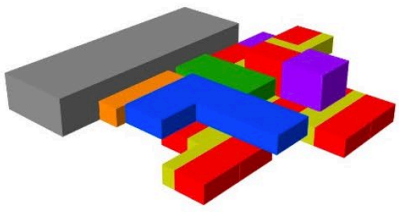


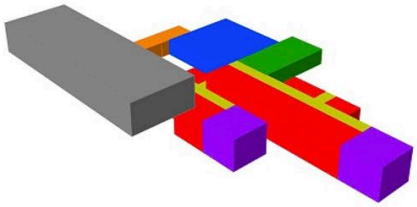


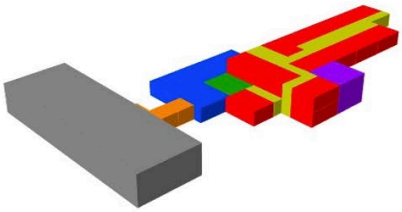


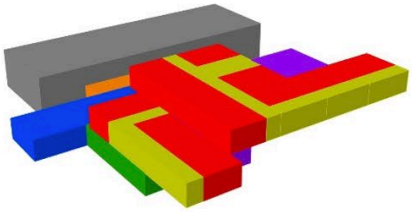


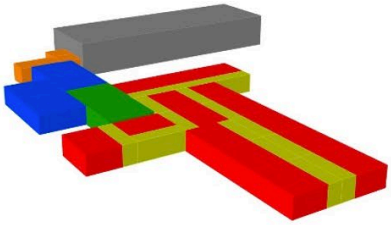


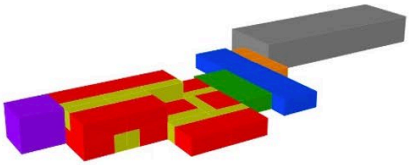


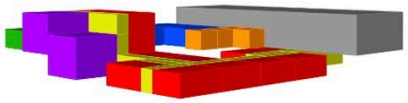


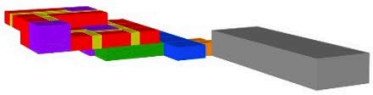


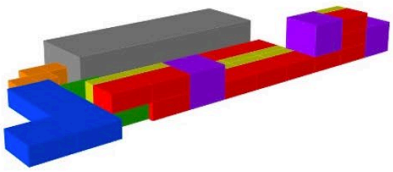


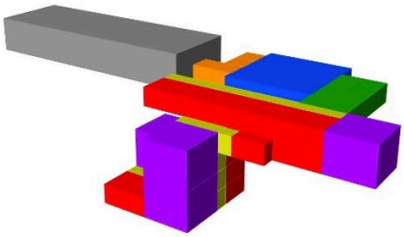


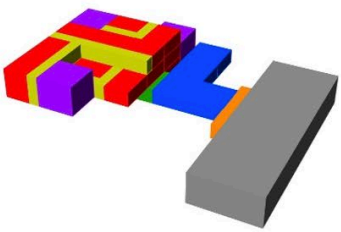


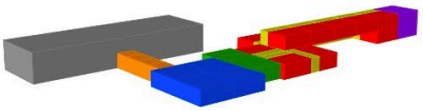


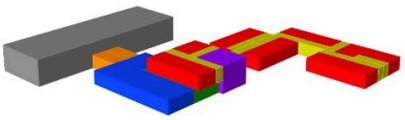


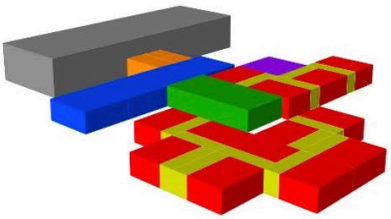


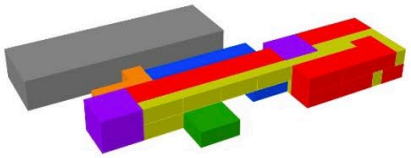


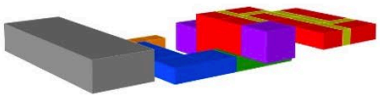


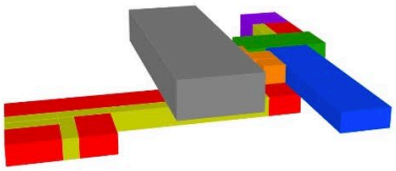


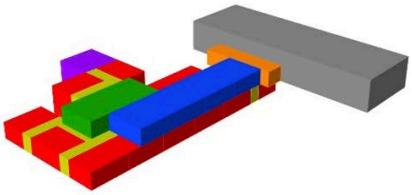


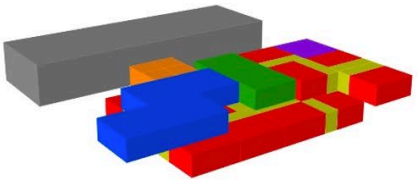


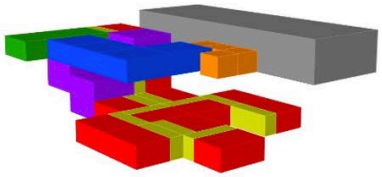


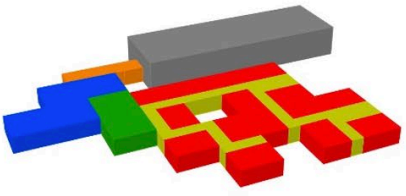


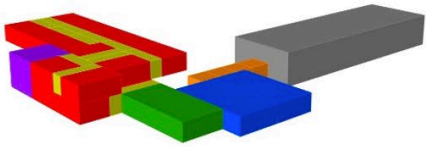


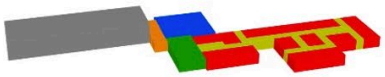


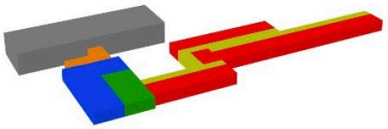


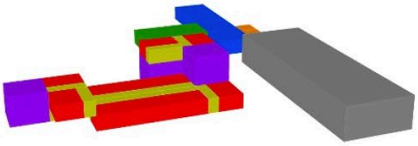


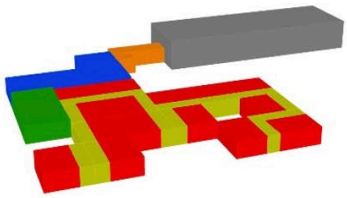


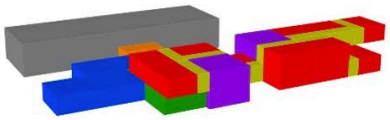


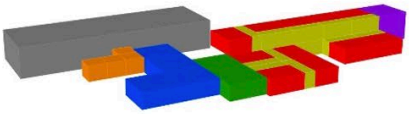


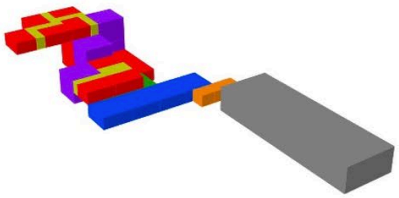


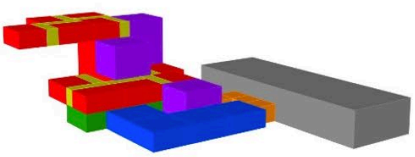


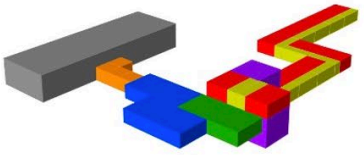


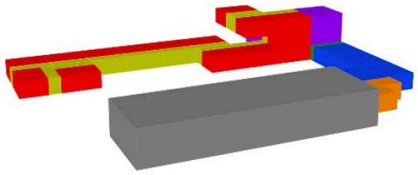


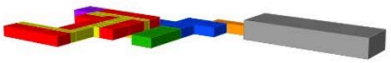


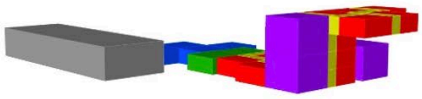


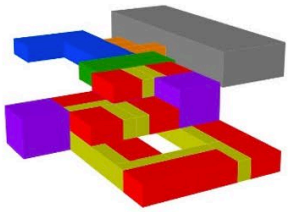


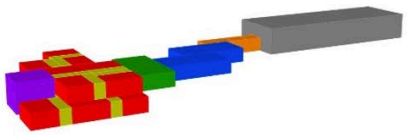


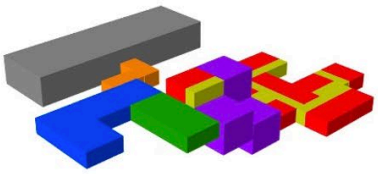


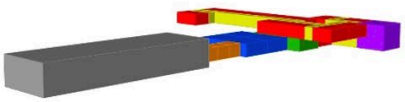


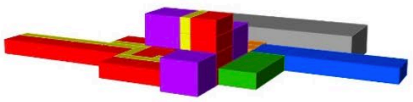


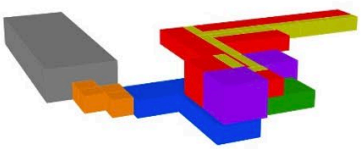


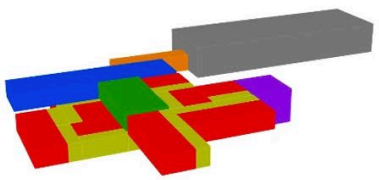


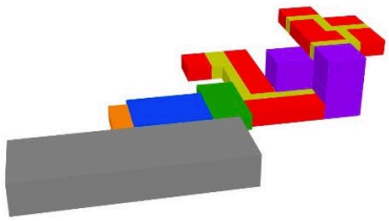


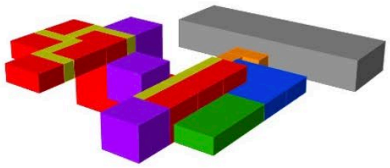


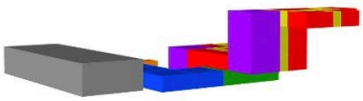


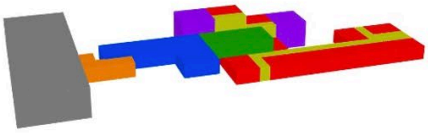


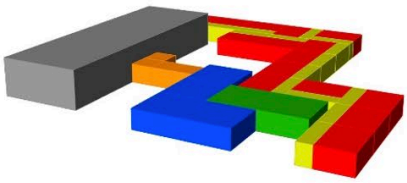


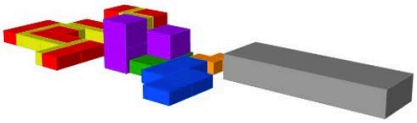


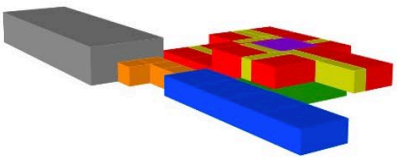


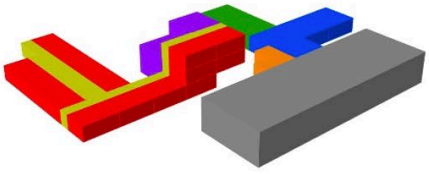


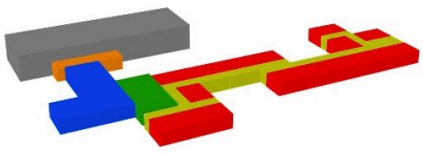


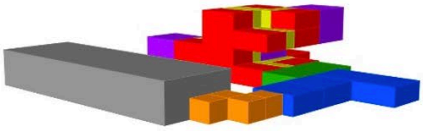


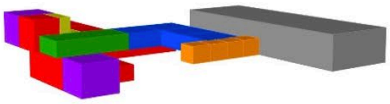


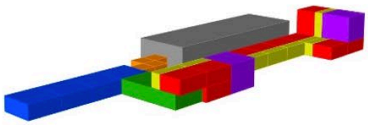


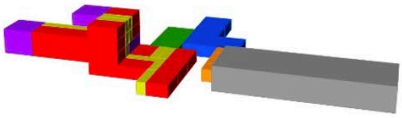


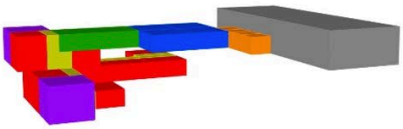


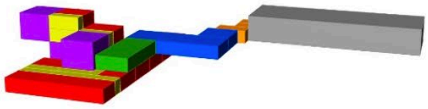


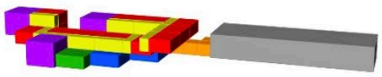


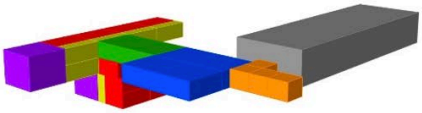


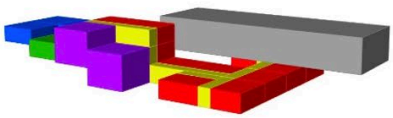


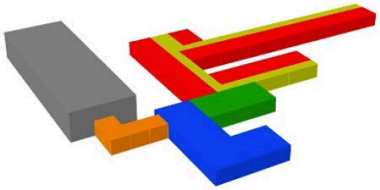


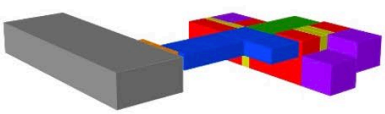


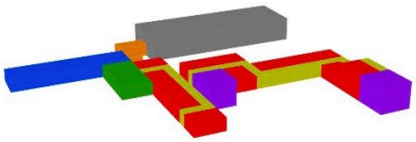


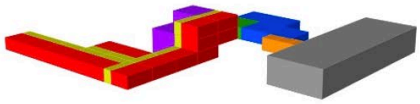


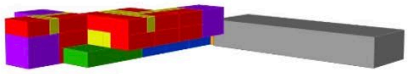


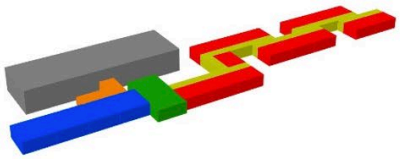


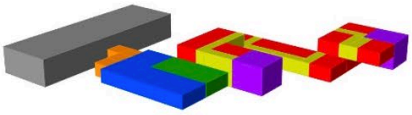


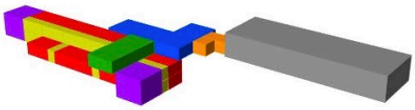


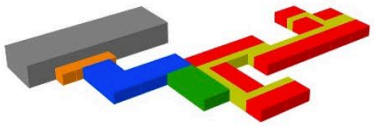


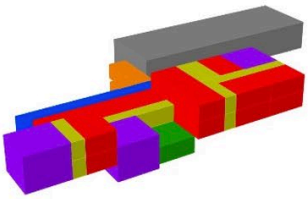


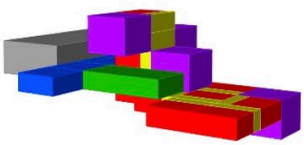


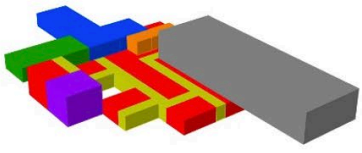


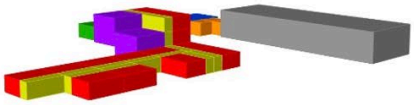


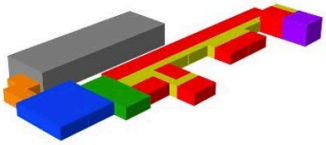


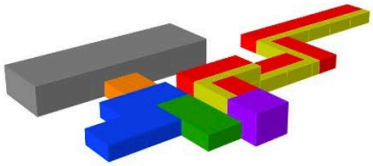


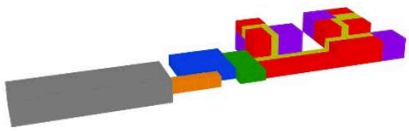


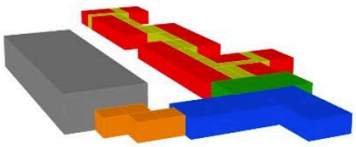


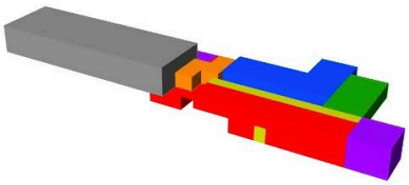


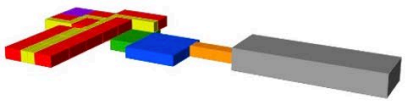


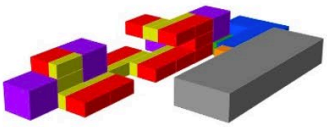


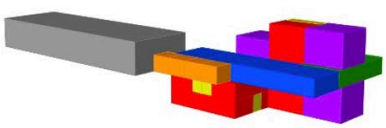


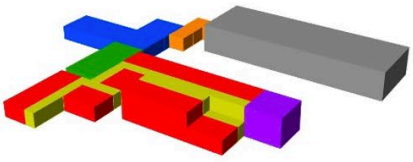




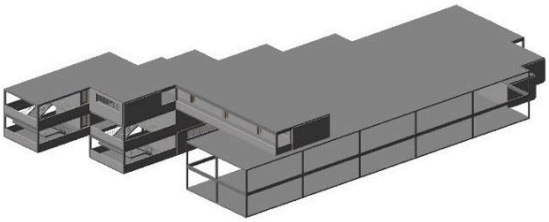
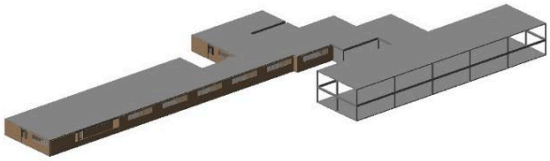
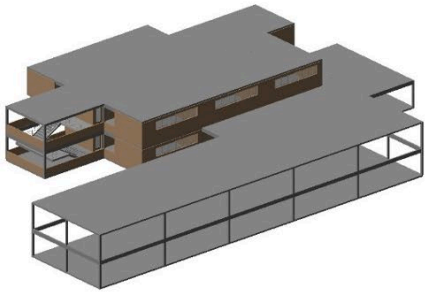
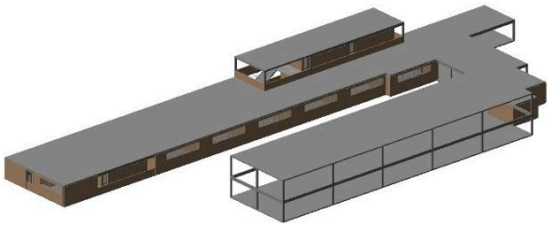
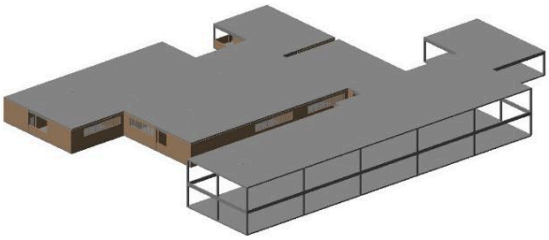
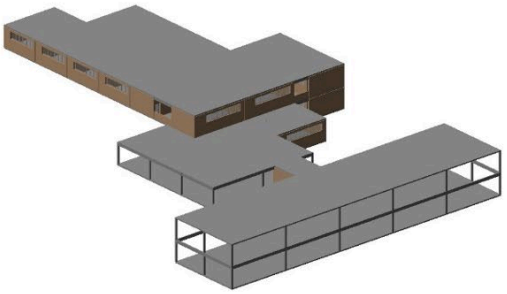
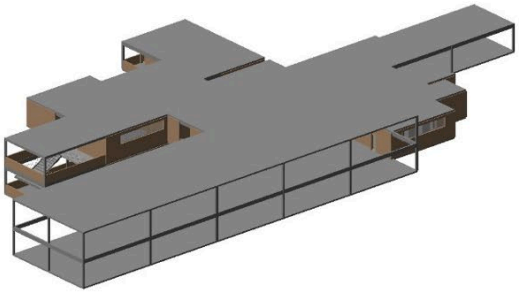
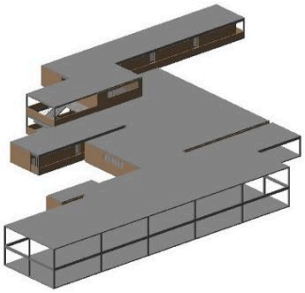
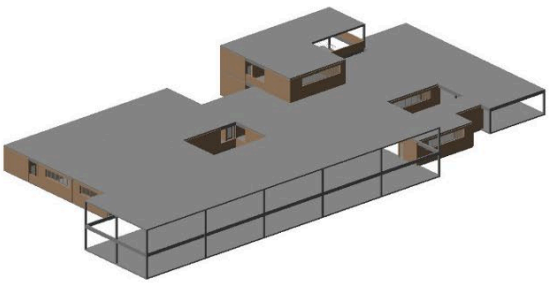
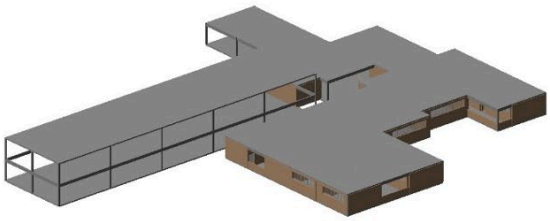


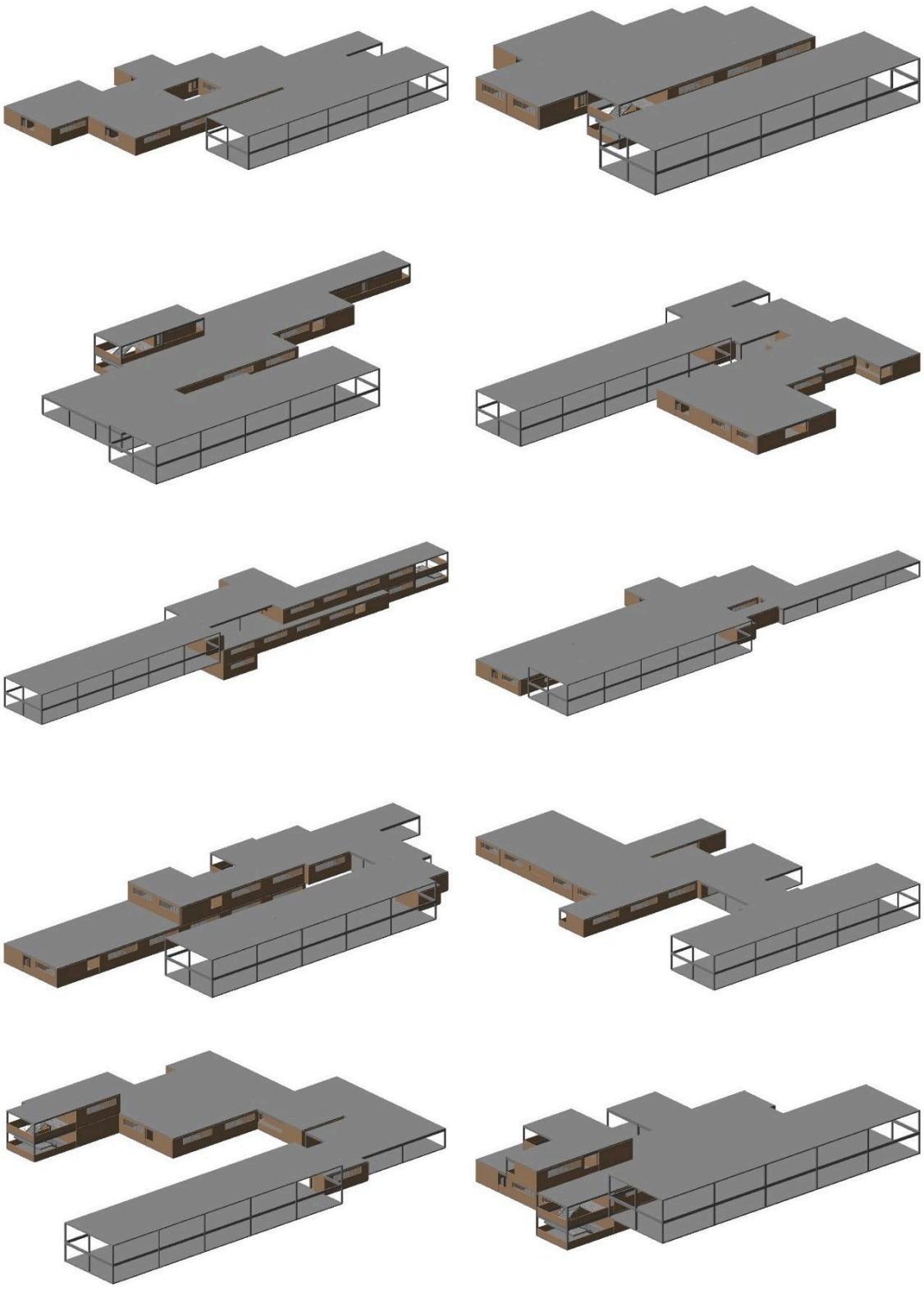


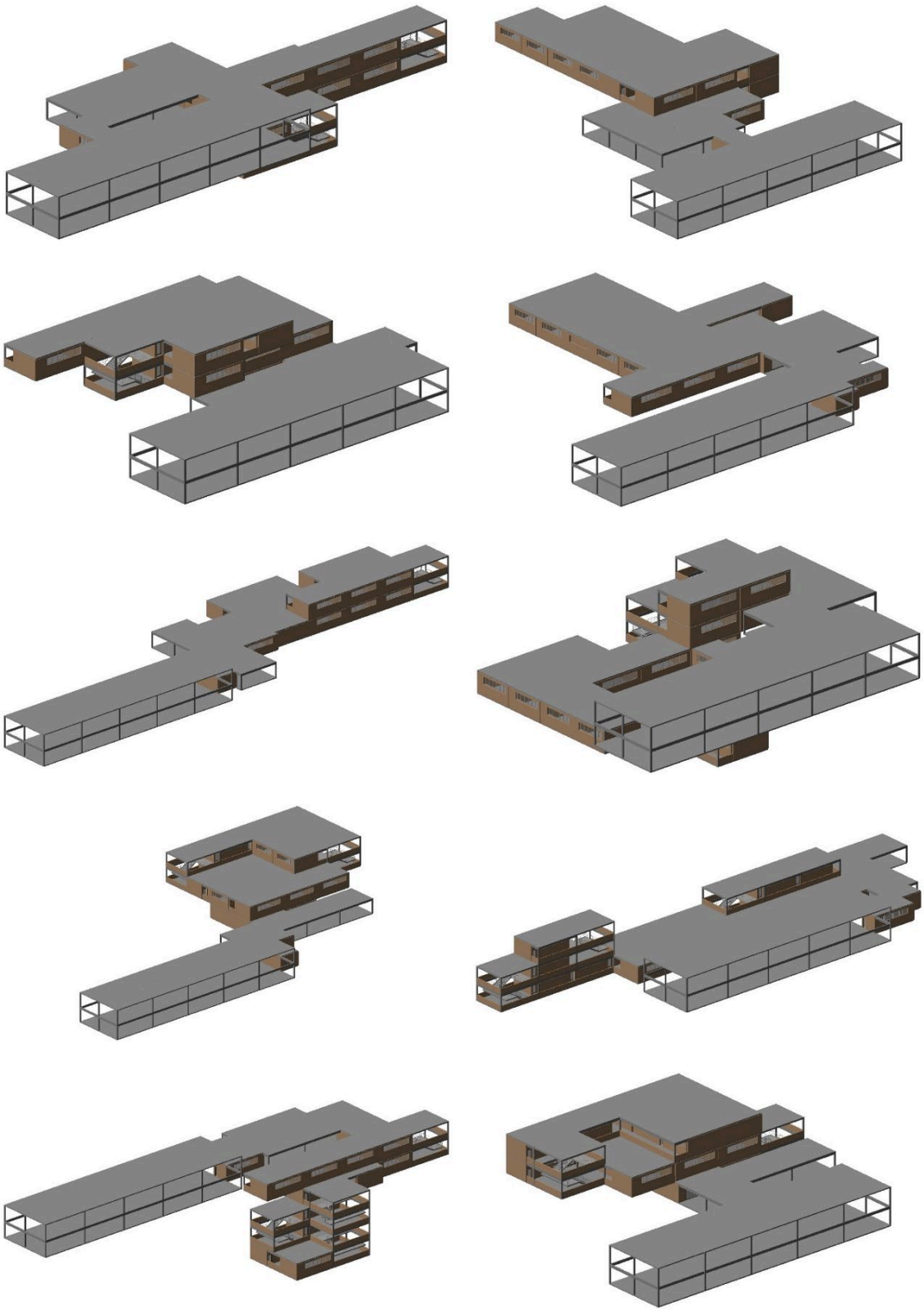


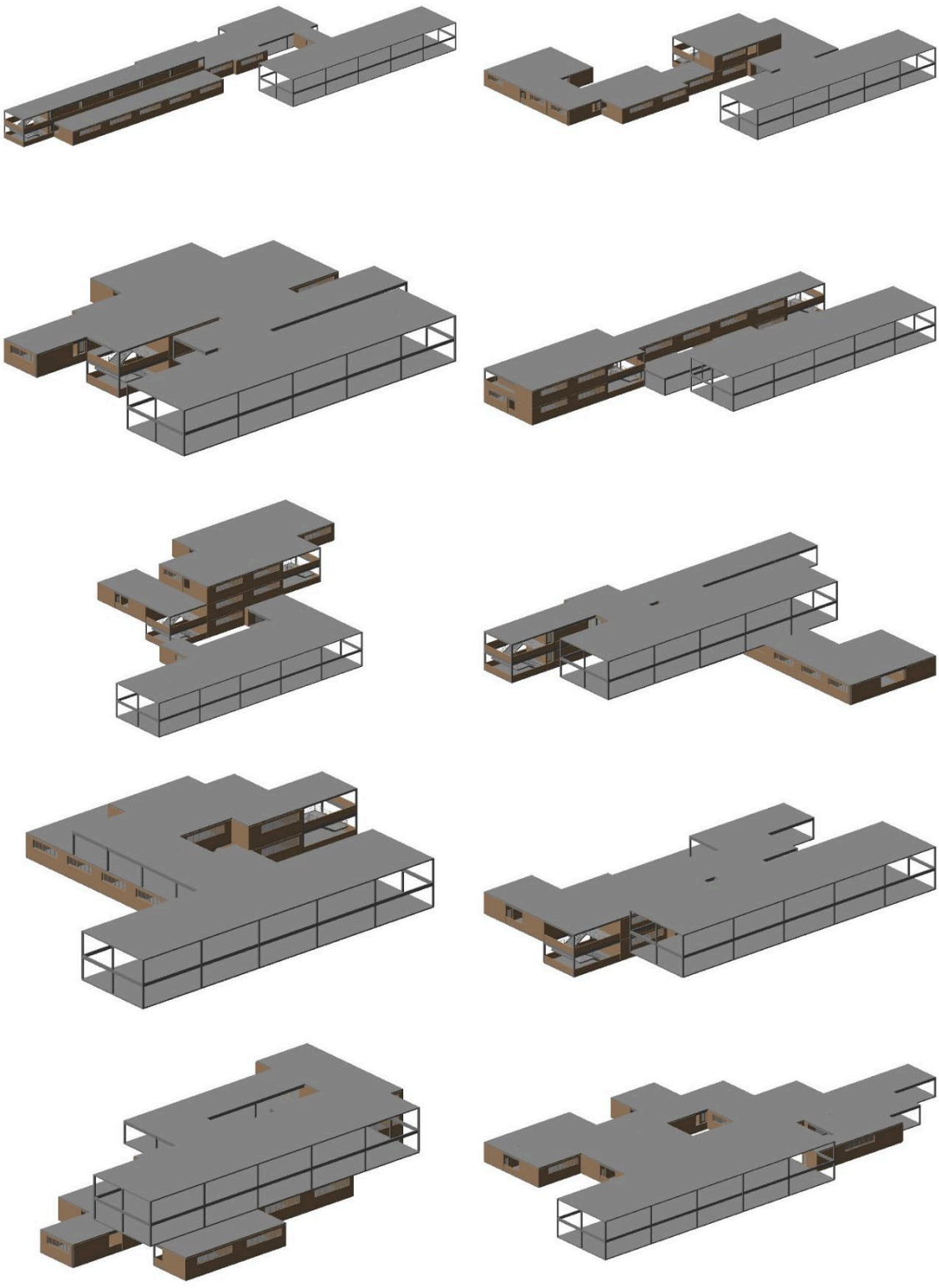


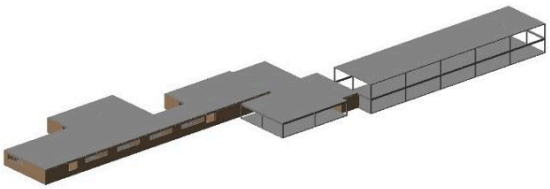
APÊNDICE C – Modelagem BIM IFC da amostragem

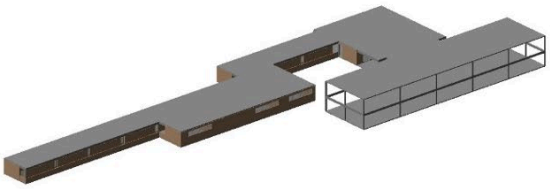











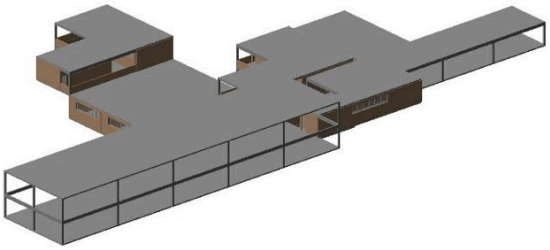


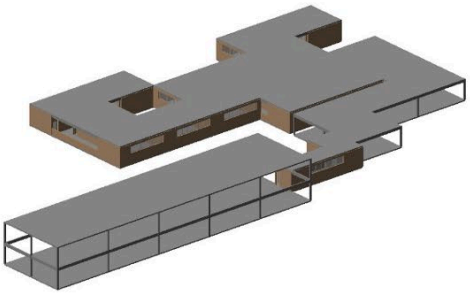


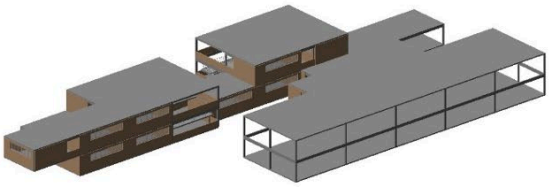


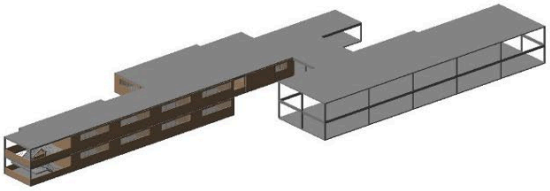


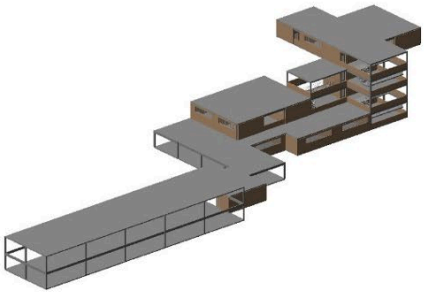


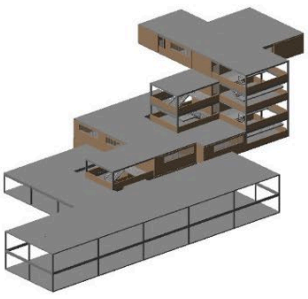


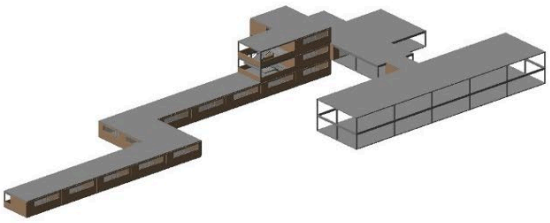


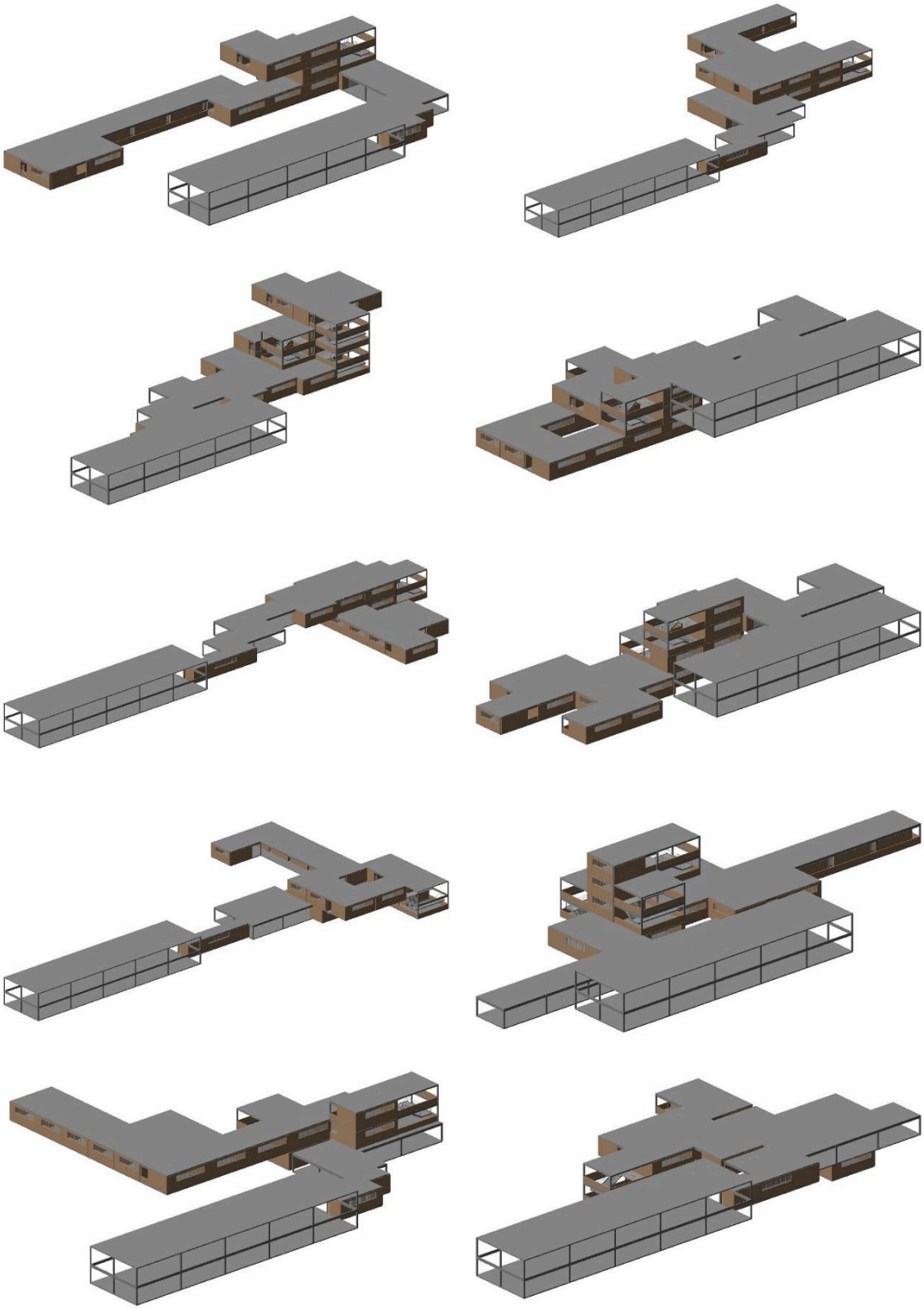


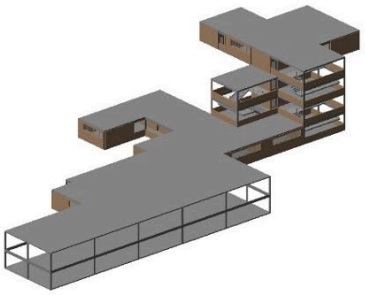
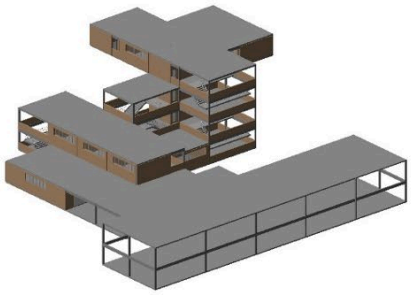
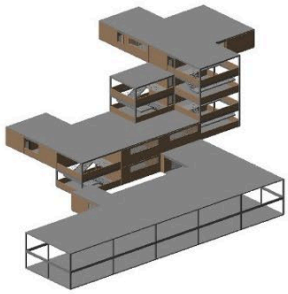
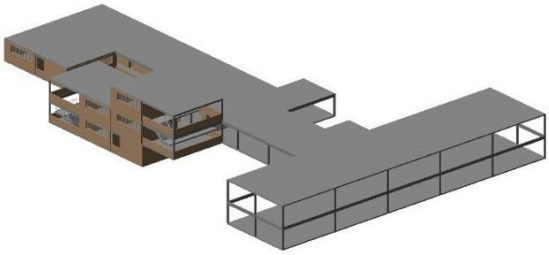
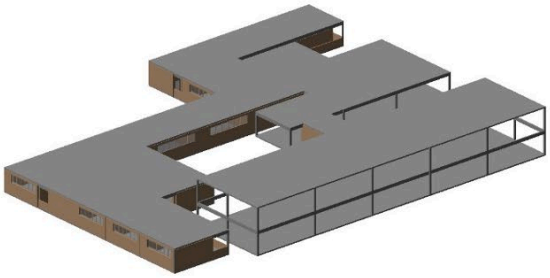
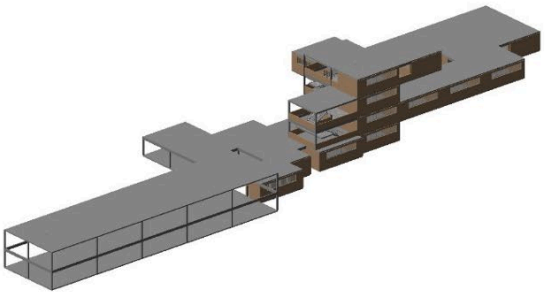
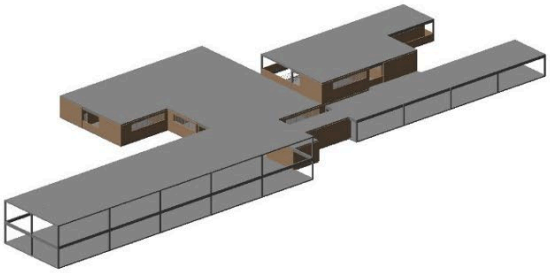
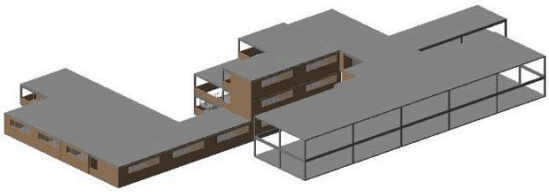
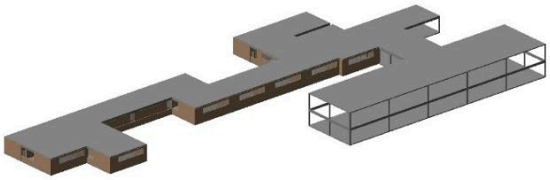
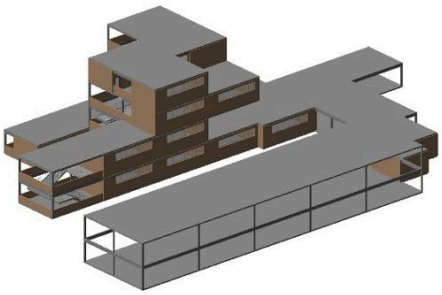


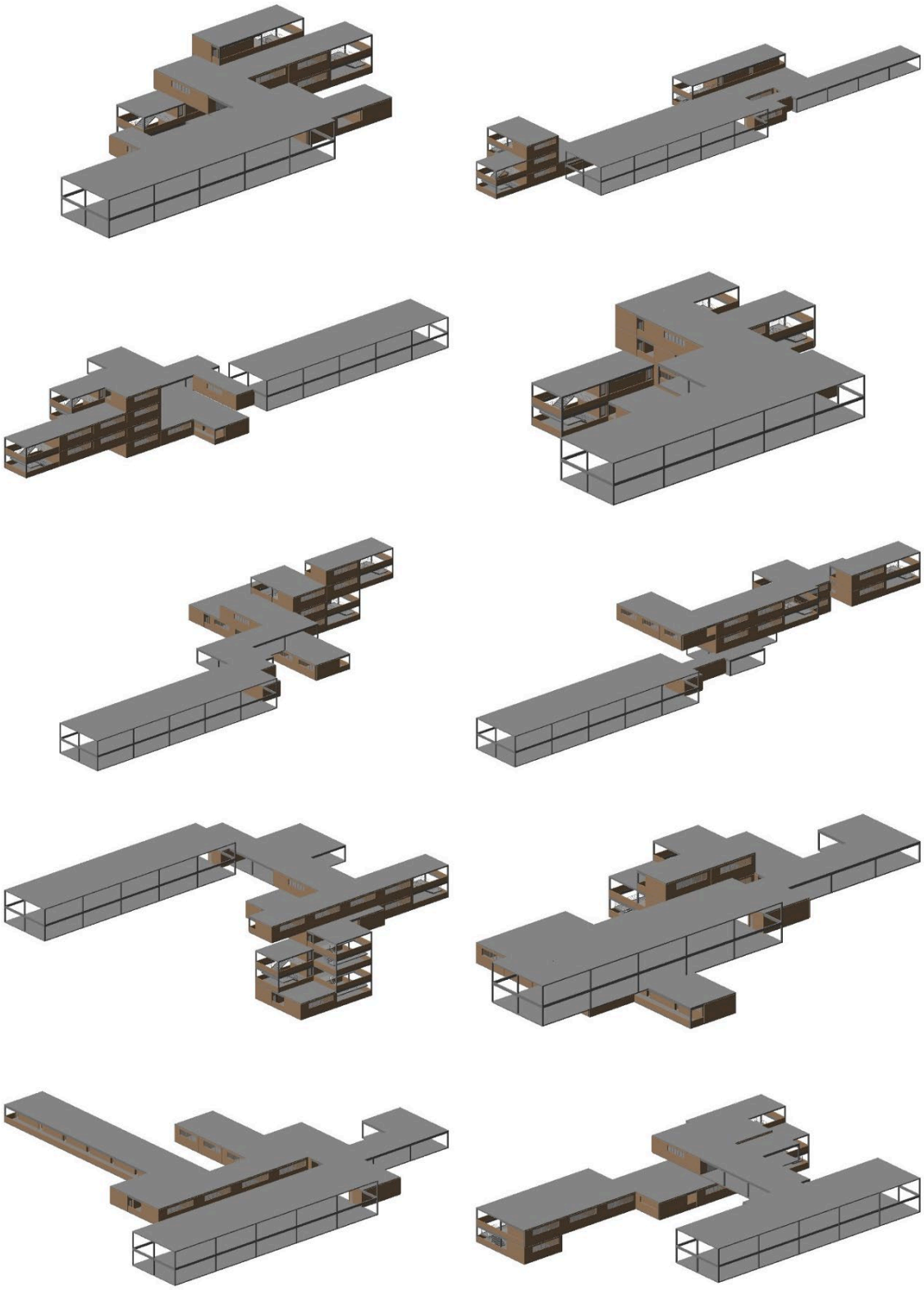


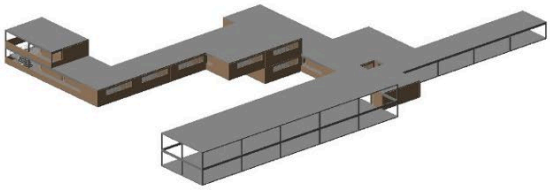
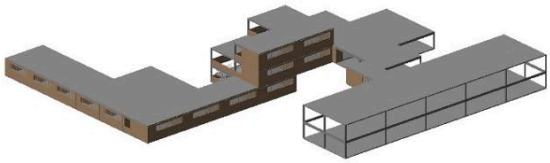
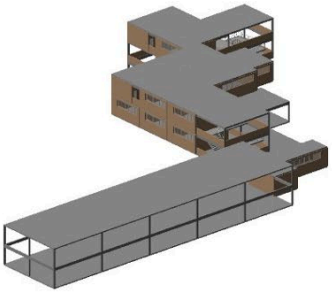
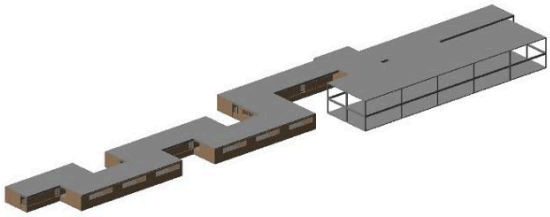
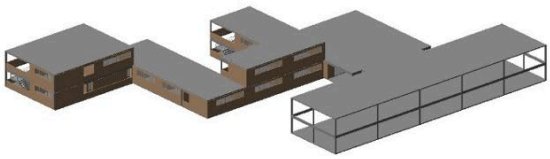
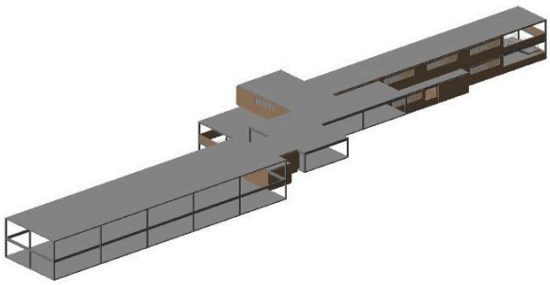
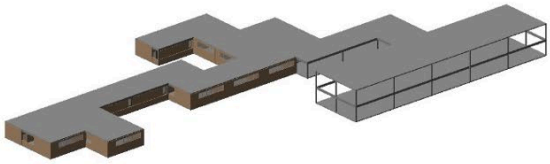
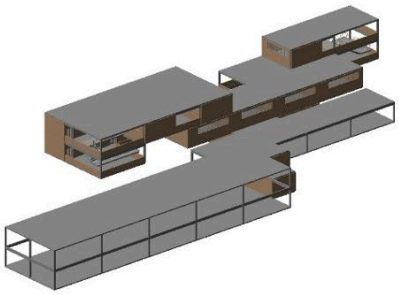
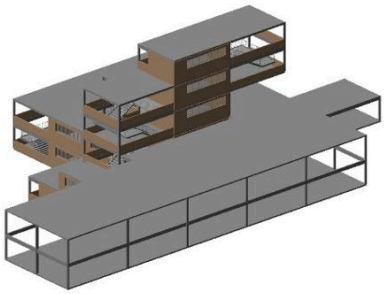


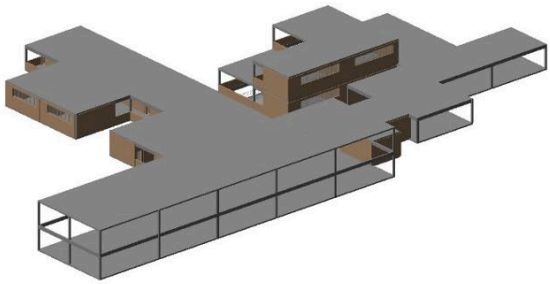


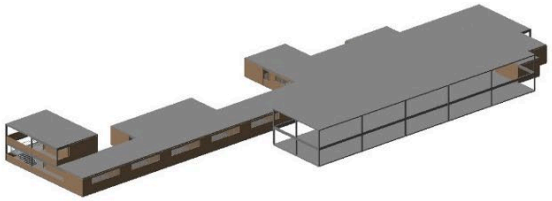


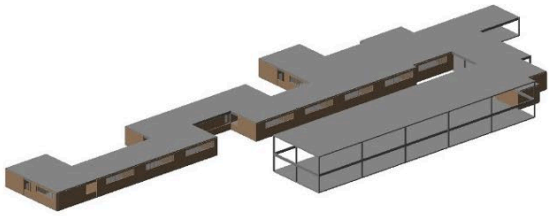











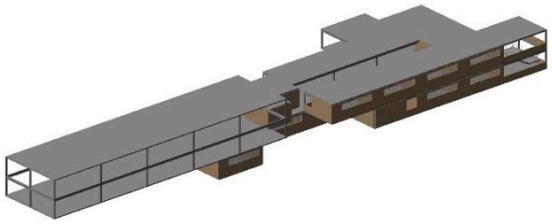


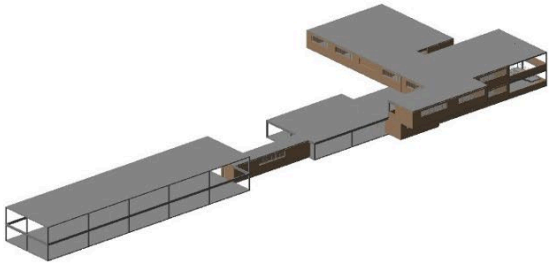










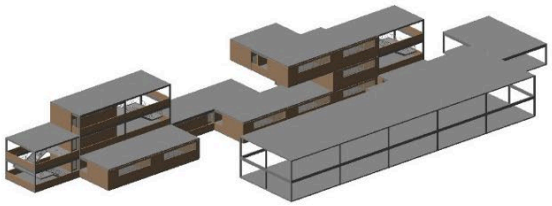



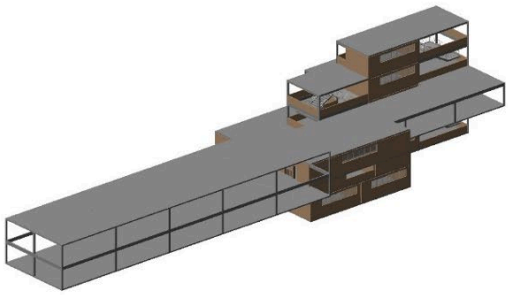


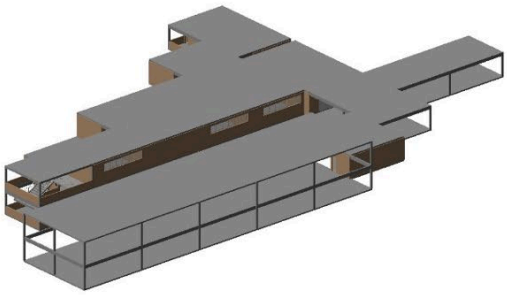












APÊNDICE D – Relatórios típicos da avaliação ciclo “Avaliar”

SOLIBRI
A NEMETSCHEK COMPANY

Analise do Projeto Conceitual

Model Name	TAG1 Version: 9.10
Checker	denise@auroraarquitetura.com
Organization	Aurora Arquitetura
Date	October 8, 2020
TAG1	Date: 2020-10-07 17:25:46 Application: ARCHICAD-64 IFC: IFC2X3

RuleSet	Rule	Description	Rule Support Tag	All Issues	Rejected Issues	Accepted Issues	Critical Issue	Moderate Issues	Low Severity Issues	Checked Components	Passed Components	Failed Components	Accepted Components	Rejected Components	Components Without Decision	Components in Critical Issues	Components in Moderate Issues	Components in Low Severity Issues
Analise do Projeto Conceitual		Enter the description here		23	0	0	**6**	**13**	**4**	1877	1862	15	0	0	15	6	13	4
1_Validacao das Circulacoes		Enter the description here		21	0	0	6	11	4	1877	1863	14	0	0	14	6	12	4
1_fluxos diretos		The rule checks that distances between specified start and destination spaces follow a given distance requirements (maximum and minimum distances). The Rule creates an issue for those spaces of matching space identification but not matching set requirements.		1	0	0	0	1	0	1877	1875	2	0	0	2	0	2	0
	fluxo para exterior	This rule checks the immediate connections between chosen spaces and if they are immediately connected to the outside of the building.	SOL/241/1.	0	0	0	0	0	0	1877	1877	0	0	0	0	0	0	0
	corredor livre	Clear width of an accessible route shall be 36 inches (915 mm).	SOL/209/1.	1	0	0	0	1	0	15	13	2	0	0	2	0	2	0
	becos sem saidas	The space required for a wheelchair to make a 180 ° turn shall be not less than 2 000 mm in the direction of travel and not less than 1 500 mm wide. e.g. If one corridor has a dead end.	SOL/209/1.	0	0	0	0	0	0	15	15	0	0	0	0	0	0	0
2_distancia a escada		The rule checks that distances between specified start and destination spaces follow a given distance requirements (maximum and minimum distances). The Rule creates an issue for those spaces of matching space identification but not matching set requirements.		20	0	0	6	10	4	21	9	12	0	0	12	6	10	4
	rota fuga das salas	The rule checks that spaces with special requirement of location (maximum distance to space X, direct access to space Y) are located correctly.	SOL/161/3.	8	0	0	0	6	2	15	7	8	0	0	8	0	6	2
	distancia conforto a escada	The rule checks that spaces with special requirement of location (maximum distance to space X, direct access to space Y) are located correctly.	SOL/161/3.	12	0	0	6	4	2	21	9	12	0	0	12	6	4	2
2_Validacao Conforto Ambiental		Enter the description here		2	0	0	0	2	0	1877	1875	2	0	0	2	0	2	0
Afastamentos iluminacao e ventilacao				0	0	0	0	0	0	1877	1877	0	0	0	0	0	0	0
	Afastamento de janelas	This rule checks if windows, doors, openings are in valid airwells.	SOL/242/1.	0	0	0	0	0	0	1877	1877	0	0	0	0	0	0	0
Iluminacao e ventilacao		Enter the description here		2	0	0	0	2	0	42	40	2	0	0	2	0	2	0
	Salas, vestiario e refeitorio	This rule checks that the model contains components of a selected types and they all have a construction type defined.	SOL/11/4.1	0	0	0	0	0	0	42	42	0	0	0	0	0	0	0
	Iluminacao e ventilacao	This rule checks that ratio between window area and area of the space is within limits.	SOL/19/3.3	2	0	0	0	2	0	15	13	2	0	0	2	0	2	0
3_Validacao de Avaliacoes Economicas		Enter the description here		0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	Ratios janelas e areas	This rule checks and reports various figures related to floors. The rule requires either Gross Area Compartments (defined in Compartmentation View), or existence of a 'gross area space' in each floor. 'Gross area space' is a space component, which represents the gross area of the floor and it groups all other spaces in the floor.	SOL/111/1.	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

SOLIBRI
A NEMETSCHEK COMPANY

Area bruta

Model Name	TAG1 Version: 9.10
Checker	denise@auroraarquitetura.com
Organization	Aurora Arquitetura
Date	October 8, 2020
TAG1	Date: 2020-10-07 17:25:46 Application: ARCHICAD-64 IFC: IFC2X3

Área bruta total:	2.395,30

Building Elements - General	Type	Gross Area	Area	Bottom Area	Length	Height	Thickness	Volume	Count	Color
Floor Slabs	GENÉRICO - ESTRUTURAL 200	4790,6	4790,6	4790,6			16	958,12	80	

SOLIBRI

Area liquida

Model Name	TAG1 Version: 9.10
Checker	denise@auroraarquitetura.com
Organization	Aurora Arquitetura
Date	October 8, 2020
TAG1	Date: 2020-10-07 17:25:46 Application: ARCHICAD-64 IFC: IFC2X3

Área líquida total:	1.316,46

Floor	Space Usage	Total Area	Average Area	Count	Color
Pavimento Térreo	circulacao horizontal	322,71	21,51	15	
Pavimento Térreo	refeitorio	148,1	148,1	1	
Pavimento Térreo	sala	773,4	51,56	15	
Pavimento Térreo	vestiario	72,25	18,06	4	

SOLIBRI

Perimetros e paredes

Model Name	TAG1 Version: 9.10
Checker	denise@auroraarquitetura.com
Organization	Aurora Arquitetura
Date	October 8, 2020
TAG1	Date: 2020-10-07 17:25:46 Application: ARCHICAD-64 IFC: IFC2X3

Perimetro total	**412,00**
Comprimento circulacao	**107,00**
Area projecao paredes	**96,02**
Area fachada	**877,80**
Comprimento n arestas	**15,50**

Building Elements - General	Type	Gross Area	Area	Bottom Area	Bounding Box Lengt	Height	Thickness	Volume	Count	Color
Beams	Concreto 200 x 300			53,2	266			15,96	42	
Beams	Concreto Armado - Pré-fabricado 200 x 400			42,8	214			17,12	32	
Beams	GENÉRICO - ESTRUTURAL 200 x 500			79,3	396,5			39,65	30	
Columns	GENÉRICO - ESTRUTURAL 250 x 250			2,38	9,5			12,25	38	
Columns	GENÉRICO - PRÉ-FABRICADO 240 x 240			1,79	7,44			6,25	31	
External Walls	Bloco de Concreto - Estrutural 150	202,89	202,89	27,67	184,45	30,8	4,2	30,43	28	
Partition Walls	Bloco de Concreto - Estrutural 200	1608,09	1309,56	96,02	487,3	300,3	18,2	256,84	91	

SOLIBRI

Area circulacao

Model Name	TAG1 Version: 9.10
Checker	denise@auroraarquitetura.com
Organization	Aurora Arquitetura
Date	October 8, 2020
TAG1	Date: 2020-10-07 17:25:46 Application: ARCHICAD-64 IFC: IFC2X3

Área total de circulação: **322,71**

Floor	Space Usage	Total Area	Average Area	Count	Color
Pavimento Térreo	circulacao horizontal	322,71	21,51	15	

APÊNDICE E – Questionário sobre asserção do termo

Section 1 of 3

Questionário de pesquisa acadêmica

Este questionário é parte integrante de uma pesquisa de Mestrado no Departamento de Engenharia de Materiais e Construção da Escola de Engenharia UFMG, na área de concentração em Tecnologia da Construção com a colaboração do laboratório Machine Intelligence and Data Science (MINDS), do Departamento de Engenharia Elétrica da UFMG.

Agradecemos sua importante colaboração!

Email address *

Valid email address

This form is collecting email addresses. Change settings

Qual a área que mais se relaciona à sua atuação? *

- ◯ Construção Civil
- ◯ Ciências da Computação

Concordo em participar espontaneamente da referida pesquisa e manter sigilo do conteúdo até sua publicação. *

- ◯ Sim
- ◯ Não

Section 2 of 3

Questão única

Para responder à questão da pesquisa, considere os conceitos a seguir sobre “Algoritmos Evolucionários” (originado na Ciência da Computação) e “Building Information Modelign - BIM” (da Construção Civil):

A Computação Evolucionária também é compreendida como uma área da inteligência computacional ou inteligência artificial, destinada à resolução de problemas de otimização através do uso de processos iterativos e tem como objetivo desenvolver, avaliar e aplicar técnicas na criação de algoritmos inteligentes (ASHLOCK, 2006). A terminologia contemporânea denota que os algoritmos envolvidos na computação e programação evolucionária são denominados Algoritmos Evolucionários, que empregam princípios semelhantes ao da seleção natural para busca de soluções otimizadas (EIBEN & SMITH, 2003).

Através da Modelagem da Informação da Construção, ou Buildign Information Modelign (BIM), um modelo tridimensional preciso do edifício é construído digitalmente com o uso de informações paramétricas. A principal característica do BIM que o distingue das tecnologias de projeto que o precedem não é a modelagem tridimensional, mas a informação sistematizada, que pode ser organizada, definida e permutável (SMITH e TARDIF, 2009).

Diante a estes conceitos iniciais considere agora a definição de “BIM Evolucionário”:

"BIM Evolucionário":

Método aplicado a projetos de edificações, no qual modelos virtuais paramétricos são gerados, avaliados e otimizados pela integração de algoritmos generativos e evolucionários. Explora a aplicabilidade dos algoritmos de inteligência artificial no fluxo de desenvolvimento e automação destes projetos.

Na sua interpretação, a definição sobre “BIM Evolucionário”: *

- ( ) Inicialmente parece adequada, merecendo atenção para melhor avaliação em estudo mais aprofundado.
- ( ) Aparentemente não demonstra nenhuma correlação com os conceitos inicialmente apresentados.

Referências bibliográficas

ASHLOCK, D. Evolutionary Computation for Modeling and Optimization. Springer, 2006.

EIBEN, A.E.; SMITH, J.E. Introduction to Evolutionary Computing. Berlin: Springer, 2003.

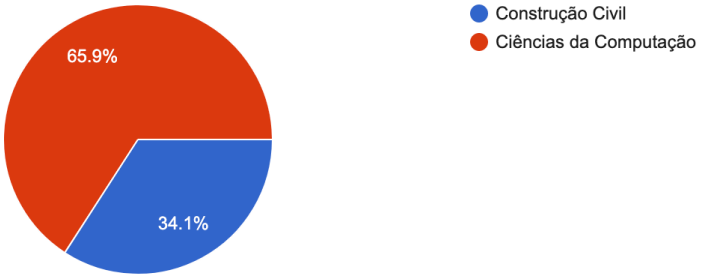
SMITH, D. K.; TARDIF, M. Building Information Modeling: A Strategic Implementation Guide for Architects, Engineers, Constructors, and Real Estate Asset Managers. Hoboken (NJ): John Wiley & Sons, 2009.

Section 3 of 3

Agradecemos sua colaboração!

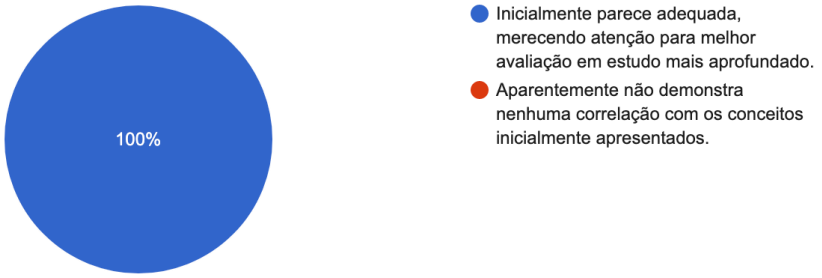
Qual a área que mais se relaciona à sua atuação?

41 responses



Na sua interpretação, a definição sobre “BIM Evolucionário”:

41 responses



ANEXO A – Lista de artigos sobre DE: 1992-2014

Autor	Ano	Artigo (título traduzido)
TAM	1992	Algoritmos genéticos, otimização de funções e design de layout de instalações
KADO	1995	Uma investigação de algoritmos genéticos para problemas de layout de instalações
JO e GERO	1995	Representação e uso do conhecimento de design no Design Evolucionário
GARCEZ-PEREZ, SCHOENEFELD e WAINWRIGHT	1996	Resolvendo problemas de layout de instalação usando programação genética
MELLER e GAU	1996	O problema do layout da instalação: tendências e perspectivas recentes e emergentes
GERO e KAZAKOV	1997	Aprendizagem e reutilização de informações em problemas de planejamento de layout de espaço usando engenharia genética
DAMSKI e GERO	1997	Uma abordagem evolutiva para gerar topologias de layout de espaço baseadas em restrições
JAGIELSKI e GERO	1997	Uma abordagem de programação genética para o problema de planejamento de layout de espaço
GERO e KAZAKOV	1997	Evolução de genes de design em problemas de planejamento de layout de espaço
JO e GERO	1998	planejamento de layout de espaço usando uma abordagem evolucionária
Li e LOVE	1998	Layout de instalações em nível de local usando algoritmos genéticos
KOCHHAR, FOSTER e HERAGU	1998	Esperança: um algoritmo genético para o problema de layout de instalações em áreas desiguais
TAVAKKOLI - MOGHADDAIN e SHAYAN	1998	Projeto de layout de instalações por algoritmos genéticos
ROSENMAN e GERO	1999	Evolução de projetos por generação de estruturas de gene complexas úteis
GARZA E MAHER	1999	Casos de layout de design em evolução para satisfazer as restrições dos filhos do Feng Shui
ELEZKURTAJ e FRANK	2000	Geometria e topologia: uma interface de usuário para a evolução artificial no projeto arquitetônico

LIGGETT	2000	Layout das instalações automatizadas: passado, presente e futuro
MICHALEK e PAPALAMBROS	2002	Otimização de design interativo de layouts arquitetônicos
HARMANANI, ZOUEIN e HAJAR	2004	Um algoritmo genético paralelo para o problema de layout de site geometricamente restrito com instalações de tamanhos desiguais
HOMAYOUNI	2007	Uma abordagem de algoritmo genético para a otimização do planejamento do layout do espaço
DOULGERAKIS	2007	Programação genética + desdobramento da embriologia no planejamento automatizado de layout
BANERJEE e QUEIROZ	2008	Um modelo de design criativo usando algoritmos genéticos interativos-colaborativos
WONG e CHAN	2009	EvoArch: um algoritmo evolutivo para design de layout arquitetônico
KNECHT	2010	Gerando layouts de planta baixa com árvores k-d e algoritmos evolutivos
FLACK	2010	Evolução das plantas arquitetônicas
THAKUR e KUMARI	2010	Planejamento de layout arquitetônico usando algoritmos genéticos
DUTTA e SARTHAK	2011	Planejamento de espaço arquitetônico usando abordagens de computação evolucionária: uma revisão
RODRIGUES, GASPAR e GOMES	2013	Uma estratégia evolutiva aprimorada com uma técnica de busca local para o problema de alocação de espaço na arquitetura, Parte 1: Metodologia
RODRIGUES, GASPAR e GOMES	2013	Uma estratégia evolutiva aprimorada com uma técnica de busca local para o problema de alocação de espaço em arquitetura, Parte 2: Validação e testes de desempenho
RODRIGUES, GASPAR e GOMES	2013	Uma abordagem para o problema de alocação de espaço em vários níveis na arquitetura usando uma técnica evolutiva híbrida
FERNANDO	2014	Planejamento de espaço e projeto preliminar usando vida artificial

Fonte: traduzido de Calixto e Celani (2015).

ANEXO B – Programa de arquitetura FDE

Programas arquitetônicos

Dezembro/2017

Anos Iniciais	M1 4 a 7 salas			M2 8 a 11 salas			M3 12 a 15 salas		
Ambientes	**Ficha**	Qtd.	Área (m²)	**Ficha**	Qtd.	Área (m²)	**Ficha**	Qtd.	Área (m²)
Direção/Administração									
Diretor	**01A**	1	9,72	**01A**	1	9,72	**01A**	1	9,72
Vice-diretor	**02A**	1	9,72	**02A**	1	9,72	**02A**	1	9,72
Secretaria	**03A**	1	19,44	**03B**	1	32,40	**03C**	1	45,36
Almoxarifado	**04A**	1	6,48	**04B**	1	12,96	**04C**	1	16,20
Coord. Pedagógico	**05A**	1	9,72	**05A**	1	9,72	**05A**	1	9,72
Professores	**06A**	1	42,12	**06A**	1	42,12	**06A**	1	42,12
Copa / Professores	**06A**	1	9,72	**06A**	1	9,72	**06A**	1	9,72
Conj. Sanit. Adm.	**07A**	1	14,58	**07B**	1	25,92	**07B**	1	25,92
Pedagógico									
Sala de aula	**08A**	4/7	51,84	**08A**	8/11	51,84	**08A**	12/15	51,84
Sala de recursos	**09A**	1	25,92	**09B**	1	51,84	**09B**	1	51,84
Uso múltiplo	**10A**	1	77,76	**10A**	1	77,76	**10A**	1	77,76
Lab. Ciências									
Sala de Preparo									
Uso Múltiplo									
Lab. Química e Biologia									
Sala de Preparo									
Lab. Matemática e Física									
Sala de leitura									
Sala de Informática	**14A**	1	51,84	**14A**	1	51,84	**14A**	1	51,84
Depósito	**15A**	1	12,96	**15A**	1	12,96	**15A**	1	12,96
Vivência									
Cozinha	**16A**	1	28,35	**16B**	1	32,40	**16B**	1	32,40
Despensa	**16A**	1	11,34	**16B**	1	19,44	**16B**	1	19,44
Refeitório	**17A**	1	72,90	**17B**	1	129,60	**17C**	1	155,52
Cantina									
Conj. sanit. alunos	**19A**	1	51,84	**19A**	1	61,56	**19A**	1	85,86
Conj. vest. alunos									
Grêmio									
Dep. Mat. Ed. Física	**21A**	1	9,72	**21A**	1	9,72	**21B**	1	12,96
Quadra coberta	**22A**	1	700,00	**22A**	1	700,00	**22A**	1	700,00
Quadra descoberta									
Espaço multiesportivo*							**22C**	1	160,00
Espaço de Convivência**	**23A**	1	200,00	**23A**	1	200,00	**23A**	1	200,00
Pátio coberto	**24A**	1	129,60	**24A**	1	194,40	**24A**	1	259,20
Serviços									
Dep. Mat. Limpeza	**25A**	1	6,48	**25B**	1	9,72	**25B**	1	9,72
Conj. Sanit. Func.	**26A**	1	12,96	**26A**	1	12,96	**26A**	1	12,96

Número de salas de aula	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**	**13**	**14**	**15**
Sub-total	690,93	742,77	794,61	846,45	1.036,80	1.088,64	1.140,48	1.192,32	1.313,82	1.365,66	1.404,54	1.469,34
Área de circulação	**30%**	**30%**	**30%**	**30%**	**30%**	**30%**	**30%**	**30%**	**30%**	**30%**	**30%**	**30%**
	207,28	**222,83**	**238,38**	**253,94**	**311,04**	**326,59**	**342,14**	**357,70**	**394,15**	**409,70**	**421,36**	**440,80**
Pátio Cob. + Quadra cob.	829,60	829,60	829,60	829,60	894,40	894,40	894,40	894,40	959,20	959,20	959,20	959,20
Área total construída	**1.727,81**	**1.795,20**	**1.862,59**	**1.929,99**	**2.242,24**	**2.309,63**	**2.377,02**	**2.444,42**	**2.667,17**	**2.734,56**	**2.785,10**	**2.869,34**

* Em prédios com capacidades de 12 a 15 salas, poderá ser implantado, à critério de projeto, espaço multiesportivo para os Anos Iniciais ou quadra descoberta para os Anos Finais e Ensino Médio.

** A área sugerida para o espaço de convivência deve ser considerada como área descoberta e poderá sofrer alteração à critério de projeto.

Fonte: FDE (2019a).

ANEXO C – Projetos FDE: requisitos e recomendações precedentes

Parâmetros:
1: Exemplo de EDF com corredor de sala de aula com carga dupla2: Exemplo precedente de quatro salas de aula agrupadas em torno da área de estudo comum (aumento do espaço de aprendizado) 3: Exemplo de EDF de salas de aula sem acesso à área externa 4: Exemplo precedente de salas de aula com acesso a espaço de ensino externo (parâmetro de transparência)
Requisitos gerais da FDE:
Distribuição vertical e horizontal de espaços a serem organizados racionalmente
Soluções compactas para minimizar o movimento da terra
Adote o design modular
Os fluxos de circulação devem ser retos e diretos, enfatizar o eixo, evitar obstáculos e becos sem saída
Reduzir áreas de circulação (máx. 30% da área total construída, excluindo ginásio e área de recreação coberta)
Áreas de circulação com boa iluminação, preferência pela iluminação natural de escadas e extremidades do corredor
Largura mínima do corredor de acordo com os códigos de segurança contra incêndio e acessibilidade. A distância de qualquer ponto do nível do piso até a escada não deve exceder 25m
Distribuição do espaço de circulação a ser ajustado às dimensões modulares estruturais da FDE
Janelas FDE padrão
Salas de aula (7,20x7,20m) A = 51,84 m^2 Mín. área da janela = 1/10 da área do piso para iluminação e 1/5 da área do piso para ventilação. Isolamento acústico para privacidade. Aberturas em paredes opostas (ventilação cruzada)
Recomendações funcionais gerais do DQI e estilos e modalidades de ensino:
Bom zoneamento para minimizar interferências de ruído
Padrões separados de circulação de alunos e visitantes
Criação de um conceito de "escola dentro de uma escola"
Zoneamento de acordo com diferentes faixas etárias
Espaços de aprendizagem (alcovas) nas áreas de circulação (corredores)
Áreas internas e externas bem relacionadas
Todos os espaços são de tamanho e forma adequados para as funções pretendidas
Os espaços de ensino e aprendizagem são adequados e adequados ao currículo e organização específicos da escola.
Espaço para estudo independente
Espaço para tutoria de pares
Espaço para trabalho em equipe colaborativo em pequenos e médios grupos (2-6 alunos)
Espaço suficiente para a aprendizagem baseada em projetos
Espaços para aprendizado baseado em tecnologia com laptops, tablets, etc.
Espaços para acesso sem fio para permitir pesquisa on-line em toda a escola
Áreas para contar histórias (assentos no chão)
Espaços ao ar livre para a aprendizagem naturalista (aprender com a natureza)
Espaço externo suficiente para interação social e recreação entre os alunos e a comunidade (uso de fim de semana nas instalações da escola)
Faça bom uso do espaço disponível
Pequenas áreas descentralizadas de alimentos
Layout para flexibilidade e padrões diários de uso do espaço
Boa supervisão do corredor (espaços de circulação)
Design com ventilação natural aprimorada
Design com iluminação natural aprimorada
Design com boas condições acústicas
Design com melhores condições térmicas
Controle excessivo de radiação solar
Design baseado em princípios de sustentabilidade
Boa transparência interior-exterior
Boa qualidade do ar interior, fresca e agradável ao longo do dia, utilizando meios passivos
Projeto baseado em formas e materiais para fornecer um conceito geral geral de construção agradável
Design que permite interações sociais, com um ambiente interno que oferece variedade, apóia a concentração e incentiva o respeito
Design com elementos que o diferenciam

Fonte: Pereira *et al.* (2018).

ANEXO D – Lista templates do Solibri Model Checker

SOLIBRI MODEL CHECKER 9.6
LISTA COMPLETA - TEMPLATES DE REGRAS

ID	Nome ENG	Nome POR	Descr
SOL 1	General Intersection Rule	Regra Geral de Intersecções	Esta regra checa a intersecção entre componentes. O usuário pode configurar quais componentes a regra irá checar e como isto será feito.
SOL 11	Model Should Have Components	Componentes Necessários no Modelo	Esta regra checa se o modelo contém componentes de tipos selecionados. Também checa se os componentes possuem um tipo de construção.
SOL 111	Floor and Gross Area Analysis	Análise de Área do Andar e Área Total	Esta regra checa e lista várias informações relacionadas aos andares. A regra requer tanto Compartimentos de Áreas Totais (definidos na Vista de Compartimentação), ou existência de um 'espaço de área total' em cada andar. O 'Espaço de área total' é um componente de espaço que representa a área total do andar e agrupa todos os outros espaços do andar.
SOL 132	Space Area	Áreas de Espaços	Esta regra checa se a área de espaços específicos está dentro de limites predefinidos
SOL 161	Distances Between Spaces	Distâncias Entre Espaços	Esta regra checa se as distancias entre espaços correspondem aos requisitos dados.
SOL 162	Spaces Must Be Included in Space Groups	Espaços Devem Estar Incluídos em Grupos de Espaços	Esta regra checa se todos os espaços do modelo estão incluídos em algum grupo de espaços.
SOL 17	Layer of Component Must Be from Agreed List	Layers dos Componentes Devem Ser de uma Lista Definida	Esta regra checa se os componentes estão atribuidos aos layers corretos.
SOL 171	Component Property Values Must Be Consistent	Medidas de Componentes Devem ser Consistentes	Esta regra checa se os componentes com o mesmo tipo construtivo em todo o modelo ou no mesmo andar possuem as mesmas dimensões
SOL 172	Fire Walls Must Have Correct Wall, Door, and Window Types	Paredes Corta-Fogo Devem Possuir Tipos Corretos de	Esta regra checa se todas as paredes entre diferentes zonas de incêndio possuem o tipo correto e que as portas e janelas nestas paredes são resistentes às chamas. Também checa se os tipos de paredes, portas e janelas corta-fogo não são usadas em outras partes do projeto.
SOL 175	Space Group Containment	Conteúdo de Grupos de Espaços	Esta regra checa se todos os grupos de espaços possuem uma quantidade requerida de determinados tipos de espaços.
SOL 176	Model Structure	Estrutura do Modelo	Esta regra checa se o modelo inclui um edifício e andares com nomes únicos. Também checa se todos os componentes estão contidos em algum andar e se todos os andares possuem componentes. E finalmente também checa se todas as janelas e portas estão contidas em paredes.

ID	Nome ENG	Nome POR	Descr
SOL 179	Escape Route Analysis	Análise de Rota de Saída	Esta regra checa se é possível sair com segurança de um edifício em caso de incêndio ou outra emergência. O edifício deve possuir uma quantidade adequada de passagens para a saída que possuam capacidade suficiente, de maneira que o tempo de evacuação não seja perigosamente longo.
SOL 19	Spaces Must Have Enough Window Area	Os Espaços Devem Possuir Área Suficiente de Janelas	Esta regra checa se a taxa entre a área de janelas e a área do espaço está dentro dos limites determinados.
SOL 190	Fire Compartment Area Must Be within Limits	Área de Compartimento de Fogo Deve Estar Dentro de Limites	Esta regra checa se as áreas de compartimentos de fogo estão dentro de limites.
SOL 191	Spaces Must Be Included in Fire Compartments	Espaços Devem Ser Incluídos em Compartimentos de Fogo	Esta regra checa se todos os espaços do modelo estão incluídos em compartimentos de fogo.
SOL 202	Space Validation	Validação de Espaços	Esta regra checa se a geometria e localização dos espaços estão corretas. Checa se as bordas dos espaços estão próximas às paredes, colunas e outros objetos, e se os espaços estão tocando uma superficie de laje acima ou abaixo. Também checa a altura dos espaços e intersecões com outros componentes.
SOL 203	Required Property Sets	Propriedades Requeridas	Esta regra checa se o modelo contém os conjunto de propriedades e as propriedades requeridas. Também checa se as propriedades possuem (ou não) um valor e se este valor é aceitável.
SOL 206	Model Comparison	Comparação de Modelos	Esta regra compara dois modelos e apresenta as diferenças entre eles.
SOL 207	Accessible Ramp Rule	Regra de Acessibilidade de Rampas	Esta regra verifica a acessibilidade de rampas sob diferentes perspectivas. Ela verifica a inclinação, comprimento, largura, e os espaços livres no início e no fim das rampas. Ela também verifica as dimensões de patamares intermediários das rampas.
SOL 208	Accessible Door Rule	Regra de Acessibilidade de Portas	Esta regra verifica a acessibilidade de portas sob diferentes perspectivas. Ela verifica as dimensões, proporção de vidro, direções de abertura e os espaços livres da porta. Para usar esta regra os espaços devem estar classificados por seu uso.
SOL 209	Free Floor Space	Espaço Livre de Circulação	Esta regra checa se os espaços possuem suficiente de circulação.
SOL 21	Components Must Have Unique Identifier	Componentes Devem Ter Identificador Único	Esta regra checa se todo componente possui um identificador único (em todo o modelo, no andar da edificação ou no mesmo grupo de espaço). Também checa se os identificadores estão corretos (quando necessário).

ID	Nome ENG	Nome POR	Descr
SOL 210	Accessible Stair Rule	Regra de Acessibilidade de Escadas	Esta regra verifica a acessibilidade de escadas sob diferentes perspectivas. Ela verifica o número de degraus, dimensões dos degraus, dimensões dos patamares intermediários, e espaços livres no início e final das escadas. Alturas de vão de passagem acima e abaixo da escada não são checadas.
SOL 211	Accessible Window Rule	Regra de Acessibilidade de Janelas	Esta regra verifica a acessibilidade de janelas sob diferentes perspectivas. Atualmente ela verifica apenas a altura do peitoril. Para usar esta regra os espaços devem estar classificados por seu uso.
SOL 212	Building Envelope Validation	Validação do Envelope da Edificação	Esta regra checa se o envelope da edificação definido no modelo (ref. Propriedade de Envelope da Edificação na vista de info de paredes) é o mesmo que o envelope de contorno dos espaços de áreas totais e/ou espaços do modelo.
SOL 213	Shelf Running Metre Rule	Regra de Distância Corrida de Estantes	Esta regra checa a distância corrida das estantes. A regra também gera um relatório de todos os espaços de armazenagem com seus comprimentos.
SOL 215	Alowed Profiles	Perfis Permitidos	Esta regra checa se apenas os perfis listados de vigas e colunas estão sendo usados no modelo.
SOL 216	Wall Validation	Validação de Paredes	Esta regra checa a geometria e as medidas das paredes. A regra possui requisitos de medidas de distâncias de janelas, portas, aberturas, e bordas de paredes. Pode possuir também limitações para aceitar tipos de geometrias de paredes. A direção da geometria de extrusões também pode ser limitada.
SOL 218	Element Hole Validation Rule	Regra de Validação de Aberturas	Esta regra checa se as aberturas estão em localizações válidas.
SOL 220	Bottom to Bottom Distances	Distâncias Entre Faces Inferiores	Esta regra checa distâncias verticais entre componentes.
SOL 221	Wall Distance	Distância de Paredes	Esta regra checa se as distâncias entre paredes estão em uma faixa aceitável.
SOL 222	Component Distance	Distância entre Componentes	Esta regra checa a distância entre componentes.
SOL 223	Structural Components Fit in Architectural Ones	Componentes Estruturais Estão Dentro dos Arquitetônicos	Esta regra checa se os componentes do modelo estrutural se encaixam dentro dos componentes do modelo arquitetônico.
SOL 224	Architectural Components Are Filled	Componentes Arquitetônicos Estão Preenchidos	Esta regra checa se componentes arquitetônicos estão preenchidos por componentes estruturais.
SOL 225	Number of Components in Space	Número de Componentes no Espaço	Esta regra checa se existe uma quantidade requerida de componentes dentro dos espaços.
SOL 226	Free Area in Front of Components	Área Livre em Frente aos Componentes	Esta regra checa se não há componentes bloqueando outros componentes predefinidos.
SOL 228	Floor Names Must Be Consecutive	Nomes de Andares Devem ser Sequenciais	Esta regra checa se os nomes dos andares possuem numeração e são sequenciais.
SOL 23	Components Must Touch Other Components	Componentes Devem Tocar Outros Componentes	Esta regra checa se os componentes tocam as superfícies de outros componentes acima ou abaixo deles.

ID	Nome ENG	Nome POR	Descr
SOL 230	Property Rule Template with Component Filters	Gabarito Regra de Propriedades com Filtros de Componentes	Esta regra checa apenas os componentes que passam pelos filtros da tabela "Componentes a Checar". A tabela "Requisitos" lista os requisitos para os componentes. Ambas as tabelas podem conter pelo menos um filtro.
SOL 231	Comparison Between Property Values	Comparação Entre Valores de Propriedades	Esta regra é usada para comparar os valores de duas propriedades associadas a um determinado componente.
SOL 232	Manual Checking Rule	Regra de Checagem Manual	Esta regra cria ocorrências definidas nos parâmetros da regra. Esta regra pode ser usada se existirem casos que ainda não podem ser checados pelas regras existentes.
SOL 233	Allowed Beam Intersections	Intersecções Permitidas em Vigas	Esta regra checa intersecções de vigas com outros componentes que podem, podem, entretanto, atravessa-las em uma área definida. Componentes (tipicamente tubos e dutos) que atravessam a viga na área permitida não irão gerar nenhuma ocorrência.
SOL 234	Component Inside Component	Componentes Dentro de Componentes	Esta regra checa distâncias entre as faces de componentes que estejam um dentro do outro.
SOL 25	Components Must be Connected to Spaces	Componentes Devem Estar Conectados a Espaços	Esta regra checa se componentes externos estão conectados a um espaço e componentes internos a dois espaços. Checa portas, janelas e aberturas.
SOL 36	Space Requirements	Requisitos de Espaços	Esta regra checa se o modelo contém uma quantidade determinada de espaços com um determinado tipo, nome ou número e área, por exemplo: 10 espaços de escritório com uma área de 10m2 +/- 5%.
SOL 37	Total Space Area on Each Floor	Área Total dos Espaços em Cada Andar	Esta regra checa se as áreas dos espaços em cada andar estão dentro dos limites fornecidos.
SOL 38	Space Count on Each Floor	Contagem de Espaços em Cada Andar	Esta regra checa se cada andar possui um número determinado de espaços de um certo tipo, ex.: se há 10 espaços de escritórios no piso térreo. Apenas os tipos definidos de espaços serão checados; espaços de tipos não listados são ignorados.
SOL 9	Property Values Must Be from Agreed List	Valores de Propriedades Devem Ser de Lista Selecionada	Esta regra checa se apenas os valores de propriedades pré-escolhidos estão em uso no modelo.

Fonte: Andrade e Silva (2017).

ANEXO E – Algoritmo Genético básico em Grasshopper Python

Grasshopper Python Script Editor

File Edit Tools Mode Help Test

```python
import rhinoscriptsyntax as rs
import ghpythonlib.components as lib
import random
import Rhino



def GenePool(list_len,Max_Value):
    motherList=[]
    for i in range(list_len):
        motherList.append(random.uniform(0,Max_Value))
    return motherList
mypool =GenePool(250,1)
def solution_Creator(solution_Count,Genepools):
    return (random.sample(Genepools,solution_Count))


def population(Population_count):
    population=[]
    for i in range(Population_count):
        population.append(solution_Creator(2,mypool))
    return population

# use this function if you wanna use the ghpythonlib library
def Function(First_Geo,Second_Geo,solutions):
    First_Geo= rs.coercebrep(First_Geo)
    interval_u = Rhino.Geometry.Interval(0.0, 1)
    interval_v = Rhino.Geometry.Interval(0.0, 1)
    
    First_Geo.Faces[0].SetDomain(0, interval_u)
    First_Geo.Faces[0].SetDomain(1, interval_u)
    
    circle_pt =rs.EvaluateSurface(First_Geo,solutions[0],solutions[1])
    circle = rs.AddCircle(circle_pt,Radius)
    point_List = Second_Geo
    return lib.PointInCurve(rs.coerce3dpointlist(point_List),rs.coercecurve(circle))[0]


def fitness(population,First_Geo,Second_Geo):
    fitess_values=[]
    for solution in population:
        test=sum(Function(First_Geo,Second_Geo,solution))
        fitess_values.append(test)
    return fitess_values

```

```
def Crossover(population):
    gene_one=[]
    gene_two=[]
    def divide_chunks(Listt, n):
    # looping till length Listt
        for i in range(0, len(Listt), n):
            yield Listt[i:i + n]
    if len(population)%2 != 0 :
        del population[-1]
    else:
        population
    child_one=[]
    for choromosome in population:
        gene_one.append(choromosome[0])
        gene_two.append(choromosome[1])
    #child_one = list (divide_chunks(zip(gene_one,reversed(gene_two)),1))
    #child_two = list (divide_chunks(zip(gene_two,reversed(gene_one)),1))
    child_whole = list ([i,j] for i,j in zip (gene_one,reversed(gene_two)))
    
    return child_whole


def select(pop_list,value_list,sorted_values):
    best_individuals=[]
    if len(value_list)>1:
        best_values=sorted_values[int(len(value_list)/2):]
        print best_values
        for item in best_values:
            best_individuals.append(pop_list[value_list.index(item)])
        return best_individuals
    else:
        return pop_list

```

```
def recursive(parents,best_fitness,sort):
    genes=[]
    genes.append(list(parents))
    print genes
    if len (select(parents,best_fitness,sort)) <2:
        return select(parents,best_fitness,sort)
    else:
        child = select(parents,best_fitness,sort)
        new_gene = Crossover(child)
        new_gene_fitness = fitness(new_gene,mysurfe,points)
        sort_list = fitness(new_gene,mysurfe,points)
        sort_list.sort()
        return recursive(new_gene,new_gene_fitness,sort_list)

mysurfe = rs.AddPlanarSrf(Curve)

def main():
    parents= population(180)
    best_fitness =fitness(parents,mysurfe,points)
    sort = fitness(parents,mysurfe,points)
    sort.sort()
    return recursive(parents,best_fitness,sort)


output = main()[0]
```

Fonte: Tabari (2020).