České vysoké učení technické v Praze
Fakulta stavební
Katedra betonových a zděných konstrukcí

Optimalizace tvaru a rozložení sloupových prvků

Diplomová práce

Bc. Michal Straka

Studijní program: Stavební inženýrství
Studijní obor: Konstrukce pozemních staveb
Vedoucí práce: Ing. Martin Petřík, Ph.D.

Praha, leden 2023

Vedoucí diplomové práce:
Ing. Martin Petřík, Ph.D.
Katedra betonových a zděných konstrukcí
Fakulta stavební
České vysoké učení technické v Praze
Thákurova 7
160 00 Praha 6
Česká republika

ČESKÉ VYSOKÉ UČENÍ TECHNICKÉ V PRAZE
Fakulta stavební
Thákurova 7, 166 29 Praha 6

ZADÁNÍ DIPLOMOVÉ PRÁCE

I. OSOBNÍ A STUDIJNÍ ÚDAJE

Příjmení: Straka Jméno: Michal Osobní číslo: 510747

Zadávající katedra: Katedra betonových a zděných konstrukcí - K133

Studijní program: N3607 - Stavební inženýrství

Studijní obor/specializace: 3608T008 - Konstrukce pozemních staveb

II. ÚDAJE K DIPLOMOVÉ PRÁCI

Název diplomové práce: Optimalizace tvaru a rozložení sloupových prvků

Název diplomové práce anglicky: Optimization of Shape and Layout of Column Elements

Pokyny pro vypracování:
- Popis problematiky parametrického modelování a optimalizace pomocí evolučních algoritmů
- Příklad využití parametrického modelování v kombinaci se statickou analýzou
- Vypracování parametrického modelu včetně návrhu sloupových prvků
- Vyřešení optimalizační úlohy pomocí evolučních algoritmů

Seznam doporučené literatury:
Vierlinger, Robert & Hofmann, Arne & Bollinger, Klaus. (2013). Emergent Hybrid Prefab Structures in Dwellings. 10.13140/RG.2.1.3351.8809.
Preisinger, C. (2013), Linking Structure and Parametric Geometry. Architectural Design, 83: 110-113DOI: 10.1002/ad.1564.
Deb, Kalyan & Pratap, Amrit & Agarwal, Sameer & Meyarivan, T.. (2002). A fast and elitist multiobjective genetic algorithm: NSGA-II. Evolutionary Computation, IEEE Transactions on. 6. 182 - 197. 10.1109/4235.996017.

Jméno vedoucího diplomové práce: Ing. Martin Petřík, Ph.D.

Datum zadání diplomové práce: 29.09.2022 Termín odevzdání DP v IS KOS: 09.01.2023
Údaj uveďte v souladu s datem v časovém plánu příslušného ak. roku

Podpis vedoucího práce Podpis vedoucího katedry

III. PŘEVZETÍ ZADÁNÍ

Beru na vědomí, že jsem povinen vypracovat diplomovou práci samostatně, bez cizí pomoci, s výjimkou poskytnutých konzultací. Seznam použité literatury, jiných pramenů a jmen konzultantů je nutné uvést v diplomové práci a při citování postupovat v souladu s metodickou příručkou ČVUT „Jak psát vysokoškolské závěrečné práce“ a metodickým pokynem ČVUT „O dodržování etických principů při přípravě vysokoškolských závěrečných prací“.

Datum převzetí zadání Podpis studenta(ky)

Čestné prohlášení

Prohlašuji, že jsem diplomovou práci na téma "Optimalizace tvaru a rozložení sloupových prvků" zpracoval samostatně za použití uvedené literatury a pramenů. Dále prohlašuji, že nemám závažný důvod proti užití tohoto školního díla ve smyslu § 60 zákona č. 121/2000 Sb., o právu autorském, o právech souvisejících s právem autorským a o změně některých zákonů (autorský zákon).

V Praze, leden 2023

.....
Bc. Michal Straka

Abstrakt

Diplomová práce se věnuje parametrickému návrhu a optimalizaci sloupových prvků. Moderním způsobem řeší návrh stavebních konstrukcí s využitím genetických algoritmů a jiných optimalizačních metod. Tyto pojmy jsou v úvodních kapitolách představeny a porovnány se staršími metodami návrhů staveb. Záměrem diplomové práce je najít vhodnou polohu a tvar sloupových prvků, k čemuž bylo využito několik metod optimalizace, které jsou v práci vysvětleny a porovnány. Velkou část práce tvoří vývoj vlastního algoritmu sloužícího ke generování a následné optimalizaci sloupů s využitím genetického algoritmu. Na závěr je ukázána praktická aplikace vyvinutého algoritmu v rané fázi návrhu stavby. V práci je zároveň kladen důraz na vysvětlování metod a postupů s pomocí obrázků, a tak mohou části práce sloužit jako návod k využívání parametrického modelování a optimalizačních metod.

Klíčová slova: Parametrické navrhování, optimalizace sloupů, genetické algoritmy, Grasshopper, Rhino, Active Energy Building, Karamba3D, Wallacei.

Abstract

This diploma thesis is focused on the parametric design and optimization of column elements. It uses modern methods, including genetic algorithms, to design building structures. These concepts are introduced and compared to older design methods in the beginning chapters. The goal of the diploma thesis is to find the appropriate position and shape of column elements, for which several optimization methods were used and compared. A significant portion of the work involves developing an algorithm for generating and optimizing columns using a genetic algorithm. Finally, the practical application of the developed algorithm in the early phase of construction design is demonstrated. The thesis also emphasizes the explanation of methods and procedures with the use of illustrations, making it a useful tutorial for using parametric modeling and optimization methods.

Keywords: Parametric design, column optimization, genetic algorithms, Grasshopper, Rhino, Active Energy Building, Karamba3D, Wallacei.

Poděkování

Mé poděkování patří hlavně Ing. Martinu Petříkovi Ph.D. za odborné vedení, ochotu a cenné rady při konzultacích a zpracování diplomové práce. Děkuji také prof. Ing. Petru Štemberkovi, Ph.D., D.Eng., Ing. Michaele Frantové, Ph.D. a Mgr. Yulii Khmurovske, Ph.D. jako dalším vedoucím semináře na katedře betonových konstrukcí za věcné připomínky a nápady. Rád bych také poděkoval Ing. arch. Šimonu Prokopovi za nápady a rady v počáteční fázi diplomové práce. V neposlední řadě patří mé poděkování rodině, partnerce a blízkým za velkou podporu po celou dobu mého studia.

Seznam obrázků

Seznam zkratek

B+G Bollinger+Grohmann. 6, 27, 28

BESO Bi-Directional Evolutionary Structural Optimization. 18, 20, 21, 28, 29, 67

CAD Computer aided design. 2, 5, 51

CFD Computational fluid dynamics. 7, 58, 59

DP Diplomová práce. 55

GA Genetický algoritmus. 30

GH Grasshopper. vi, 17, 18, 21, 31

GSO Generalized Shape Optimization. 18

IT Informační technologie. 2

LO Layout Optimization. 18

MKP Metoda konečných prvků. 2, 6, 7, 18, 33

NSGA-II Non-dominated Sorting Genetic Algorithm. vi, 13, 14, 63

NURBS Non-uniform rational basis spline. 5

PCP Parallel Coordinate Plot. vi, vii, 15, 16, 63, 64

SIMP Solid Isotropic Material with Penalization. vi, 17, 21, 22, 67

TO Topologická optimalizace. vi, 17, 18

Obsah

Kapitola 1

Úvod

Cílem diplomové práce je využití parametrického modelování spolu s optimalizačními metodami k řešení úlohy optimalizace polohy a tvaru sloupových prvků. Práce a problematika optimalizace sloupových prvků je inspirována budovou Active Energy Building, ve které jsou použity šikmé sloupové prvky ve tvaru A nebo V, které byly navrženy s využitím parametrického modelování a genetického algoritmu. Motivací práce je použít moderní metody návrhu a vícekriteriální optimalizaci k vytvoření funkčního algoritmu sloužícího ke generování sloupových prvků v konstrukci.

V úvodních kapitolách jsou vysvětleny pojmy parametrického navrhování a optimalizace konstrukcí spolu s historickým původem obou metod, pomocí kterého lze tyto nové metody připodobnit k jednoduchým a představitelným modelům. Zároveň jsou zde popsány použité softwary, díky kterým lze návrhy a simulace provádět v reálném čase s využitím výpočetní techniky. V kapitole optimalizace konstrukcí jsou zobrazeny první výsledky optimalizace polohy sloupů pomocí metod topologické optimalizace. V další kapitole je představena zmiňovaná referenční stavba Active Energy Building spolu s konstrukčním řešením sloupů. Následující kapitola je věnována vývoji vlastního algoritmu ke generování sloupových prvků, která je doložena postupy a grafickými návody. V poslední kapitole je ukázka praktické aplikace algoritmu v rané fázi návrhu na vytvořeném modelu budovy. Je zde také popsána tvorba modelu spolu s analýzou prostředí, statickou analýzou a přípravou optimalizace. Na závěr je provedená optimalizace vyhodnocena, včetně grafických a tabulkových výstupů.

Kapitola 2

Parametrické navrhování

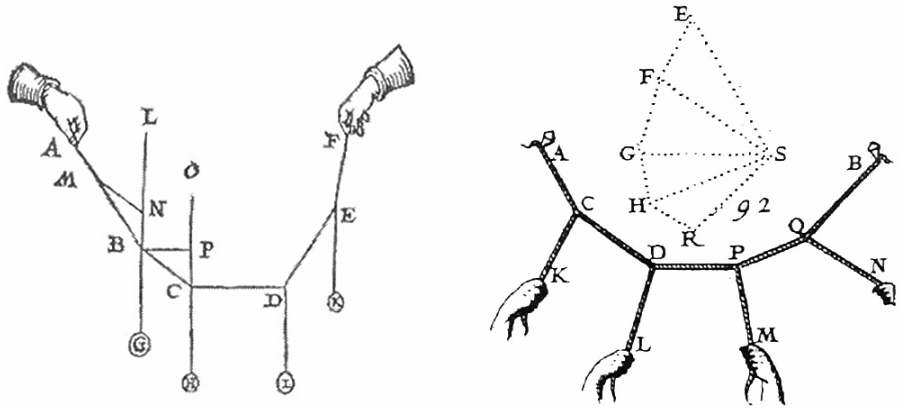
Pro řešení problematiky optimalizace nosného systému bylo nutné využít nástroje, pomocí kterých lze geometrii budovy jednoduše vytvářet a následně i měnit dle určitých optimalizačních parametrů, a to nejlépe v co nejkratším časovém úseku. To například tradiční software pro výpočty MKP nebo běžný kreslící CAD software nesplňuje. Nabízí se tak určitý protipól, a to využití programovacích jazyků, a vše si tak pomocí různých balíčků a rozšíření naprogramovat.

Určitou střední cestou nebo spojnicí mezi těmito dvěma světy je tzv. parametrické navrhování (nebo také parametrické modelování). Jedná se o metodu, kterou lze dnes připodobnit no-code nebo low-code platformám ve světě IT (např. při tvorbě webových stránek), kde uživatel přímo nepíše kód, ale využívá předdefinované funkce, které do sebe skládá a pomocí nich vytváří své aplikace. Podobně je tomu v architektuře, projektování nebo designu, a stejně jako v IT, je to metoda uživatelsky přívětivá a rychle rozvíjející. V době, kdy se návrhy staveb kreslí pomocí CAD programů, může být parametrické modelování novým přístupem k navrhování, ve kterém se místo přímé tvorby geometrie, vytváří tzv. vazby mezi geometrií a sadou pravidel či parametrů. Tyto parametry, proměnné hodnoty, si lze představit jako nitě či provázky, pomocí nichž vytváříme a kontrolujeme různorodé geometrie. Princip parametrů vychází z matematiky, kde má dlouhou historii a jedním z příkladů mohou být parametrické rovnice, např. rovnice popisující křivku zvanou řetězovka, která má velké zastoupení v historii parametrického navrhování [1].

$$y = \frac{a}{2}\left(e^{x/a} + e^{-x/a}\right) = a\,cosh\left(\frac{x}{a}\right)$$

Křivka řetězovky je popsána funkcí hyperbolického kosinu s parametrem „a“, který rovnoměrně určuje velikost křivky. Princip řetězovky ve stavitelství, tedy využití modelu volně visícího řetězu, plně taženého a jeho invertního tvaru, který

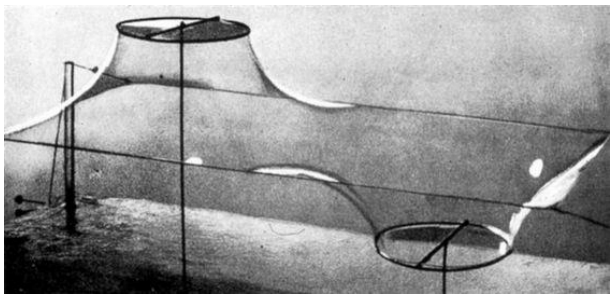
je naopak plně tlačený, se využíval již v dávné historii ke stavbě oblouků a kleneb. Prvním, kdo popsal vztah mezi řetězovkou a tlačenými oblouky byl anglický vědec, vynálezce Robert Hooke (* 1635, † 1703) větou: *„Ut pendet continuum flexile, sic stabit contiguum rigidum inversum.“* Ta se dá volně přeložit jako: „Jak volně visí lano, tak ale obráceně bude stát pevný oblouk.“ [2].



Obrázek 2.1: Schémata závěsných modelů. Autor Simon Stevin, Pierre Varignon. [2]

Éra Roberta Hooka znamenala také začátek doloženého používání závěsných modelů, které mohou být považovány jako tehdejší alternativu parametrického navrhování a počítačových modelů. Cílem závěsných modelů bylo především navržení staticky efektivního oblouku či klenby, což znamenalo minimalizovat účinky ohybového momentu či smykových sil na konstrukci. Důvodem byly především tehdejší materiály a způsob stavby oblouků a kleneb, kdy se převážně využívalo zdění z opracovaného kamene a později z cihel. Jako hlavní a přímý předchůdce parametrického modelování se uvádí až závěsné modely Antonia Gaudího (* 1852, † 1926) [3]. Katalánsky architekt upřednostňoval fyzické modely při návrhu svých budoucích staveb a závěsné modely použil při návrhu kostela v Colònii Güell nebo při návrhu slavného chrámu Sagrada Família. Gaudího model se skládal z provázku, představující konstrukce sloupů a kleneb, a zavěšených sáčků s broky, které představovaly například zatížení od věží. Provázky byly ukotveny v místech, která na modelu představovala základy pro sloupy. Přesný model v měřítku 1:10 byl zavěšený směrem dolů k zemi, tak aby bylo možné využít zmíněných principů podléhajících Newtonovo zákonům. Gaudí pak mohl s využitím nezávislých parametrů, jako byly délky provázku, poloha kotvících bodů nebo hmotnost či pozice závaží měnit tvar a v reálném čase sledovat změny konstrukce. Pomocí takového „parametrického“ modelu mohl architekt vytvořit několik variant návrhu kostela a zároveň mít ihned představu o statickém chování návrhu. Z využívání různých modelů k návrhu staveb a tvarů se ve 20. sto-

letí vyvinul princip zvaný „form finding“, přeloženo jako hledání formy, spočívající ve zkoumání tvarů a výsledků, které jsou výstupem například z již zmiňovaných modelů či experimentů. Vhodné formy a tvary jsou často inspirovány přírodou, přírodními tvary nebo přírodními experimenty. Stavby významných architektů minulého století, jako Félixe Candely v Mexiku, Piera Luigi Nerviho v Itálii nebo Freie Otta v Německu jsou příkladem principu hledání formy, kde zmiňovaní architekti našli vhodný tvar pro své stavby i pomocí fyzických modelů a zkoušek. A i k tomu slouží popisovaná metoda parametrického modelování, která v dnešní době počítačů umožňuje generování různorodých tvarů a možností konstrukcí mnohem jednodušeji než dříve.



Obrázek 2.2: Experiment Freie Otta k návrhu membránových konstrukcí[4]

Parametrismus

Prvně se o pojmu parametrické architektury zmínil italský architekt Luigi Moretti (* 1907, † 1973) ve 40. letech 20. století, který pojem vyjádřil slovy *Architettura Parametrica* v rámci svého průzkumu vztahu architektury a parametrických rovnic. Moretti byl pravděpodobně také první, kdo pomocí počítače (tehdy IBM 610) navrhl parametrický model stavby. Jednalo o návrh sportovního stadionu, který prezentoval v roce 1960 [1][3].

Postupné rozšiřovaní parametrické metody do praxe architektů v 21. století dalo vzniknout novému architektonickému stylu zvanému Parametrismus, který je často uváděn jako nástupce postmoderní architektury. Jméno architektonickému stylu dal nynější hlavní architekt studia *Zaha Hadid Architects* Patrik Schumacher v roce 2008 [3]. Právě britská architektka iráckého původu Zaha Hadid (* 1950, † 2016) je se svými stavbami považována za představitelku parametrické architektury. Parametrismus se často vyznačuje zajímavými, složitě charakterizovatelnými tvary, které vznikly digitálně pomocí funkčních parametrů. Tyto parametry mohou být víceméně jakéhokoliv rázu, pokud budou dávat logiku a budou vylepšovat funkci budovy. V mnoha dnešních architektonických kancelářích, které se věnují parametrismu, tak

působí tzv. digitální architekti, kteří se věnují této problematice, tvoří a neustále vylepšují parametrické modely. Zlepšováním modelů a posouváním technologie vpřed vznikají tak efektivní a časově úsporné návrhy staveb.

Výše napsané řádky patřily parametrickému modelování v rukou architekta. V rukou statika nebo obecněji stavebního inženýra, může parametrické modelování znamenat stejně mocný nástroj jako u architekta. Nosné systémy staveb jsou součástí architektury a návrhu stavby, po které se vyžaduje, aby plnila funkci estetickou a dnes čím dál tím častěji i funkci udržitelnou. Stavební inženýr tak může využívat své znalosti konstrukcí spolu s metodou parametrického modelování k návrhu velmi efektivních nosných systémů, které zároveň budou plnit funkce výše zmíněné. Důkazem toho můžou být například inženýři z německé společnosti Bollinger+Grohmann, kteří posouvají statiku stavebních konstrukcí kupředu využíváním nových technologií.

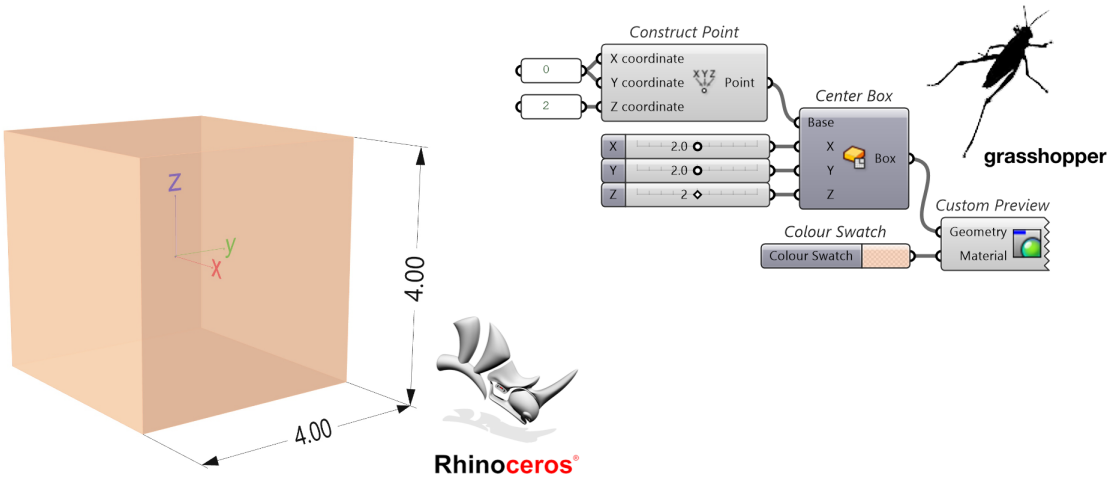
2.1 Používaný software

V dnešní době technologií a výkonné výpočetní techniky lze fyzikální modely či experimenty vytvářet a simulovat pomocí různých programů a aplikací. Například k simulaci membrán již není nutné vytvářet modely pomocí mýdlových bublin nebo natažených látek. Takový model lze jednoduše vytvořit a modifikovat na běžném počítači. Stejně tak lze provádět analýzy prostředí, jako simulace oslunění, stínění, výhledů nebo proudění vzduchu. K tomu všemu je však potřebný software, pomocí něhož lze zmíněné modely vytvářet. V následujících odstavcích proto bude popsán software, který byl využíván k řešení diplomové práce. Samostatnou kategorií je optimalizace konstrukcí a aplikace k ní využívané, které budou popsány v další kapitole 3.

2.1.1 Rhinoceros 3D & Grasshopper

Rhinoceros 3D zkráceně Rhino je grafický 3D a CAD software pro vytváření prostorových objektů. Je založen na modelu NURBS, který slouží k přesnému a matematicky popsatelnému vytváření křivek. Příkladem takové křivky a vzniku NURBS je např. Bézierova křivka popsána v podsekci 5.5.3. Program Rhino je využíván širokou škálou profesí, jako například designéry, architekty nebo strojaři.

Nedílnou součástí pro parametrické modelování je rozšíření programu nazvané Grasshopper. V prostředí Grasshopper lze totiž programovat pomocí zmíněných předdefinovaných funkcí, které se v komunitě Grasshopper nazývají komponenty. Sa-



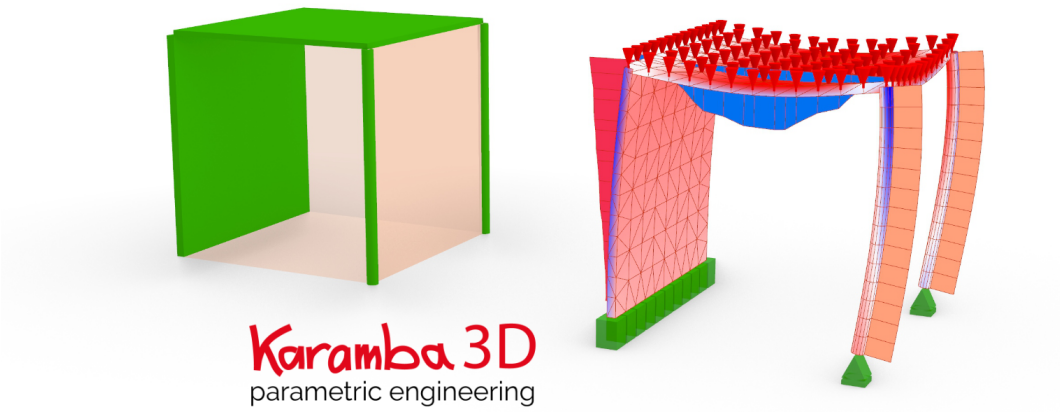
Obrázek 2.3: Ukázka použití softwaru Rhinoceros a Grasshopper

motná komponenta sestává ze vstupů a výstupů, do kterých lze zapojit další komponenty nebo proměnné parametry. O samotném programování v prostředí Grasshopper se pak spíše vyjadřuje jako o vizuálním programování nebo skriptování, které spočívá v propojování komponent, čímž se vytváří vizuální program, nazývaný definice. Spojením aplikací Grasshopper a Rhino tak vzniká vazba mezi zobrazenou geometrií v Rhino a funkční definicí v prostředí Grasshopper. Jednoduché příklady tvorby geometrie jsou také ukázány v kapitole 5.

Prostředí Grasshopper nabízí možnost přidávání rozšiřujících balíčků, které mají za úkol doplnit a přidat nové komponenty k širšímu využití programu. Rozšíření, nebo také z angličtiny pluginy, se dají chápat jako přídavné aplikace, které jsou vyvíjeny společnostmi či jedinci a můžou být jak komerčního, tak volného charakteru.

2.1.2 Karamba3D

Jedním z rozšiřujících balíčku je Karamba3D, což je program pro interaktivní analýzy konstrukcí založen na MKP. Za jeho vývojem stojí Clemens Preisinger ve spolupráci s B+G. Karamba3D byla vytvořena za účelem rychlé statické analýzy pro architekty a stavební inženýry zejména v rané fázi návrhu [5]. Hlavní výhodou oproti tradičním programům na analýzu konstrukcí je okamžitá odezva programu na změnu geometrie, což koresponduje s myšlenkou parametrismu a parametrického modelování. Navíc, rychle vypočtené výsledky tak lze implementovat do procesu optimalizace. Možnost sledovat chování konstrukce reagující na změny parametrů v reálném čase je velmi přínosné nejenom při návrhu stavby, ale i v edukační rovině. Tento způsob či princip lze připodobnit závěsným modelům, které jsou popsány v úvodu kapitoly.



Obrázek 2.4: Ukázka použití softwaru Karamba3D

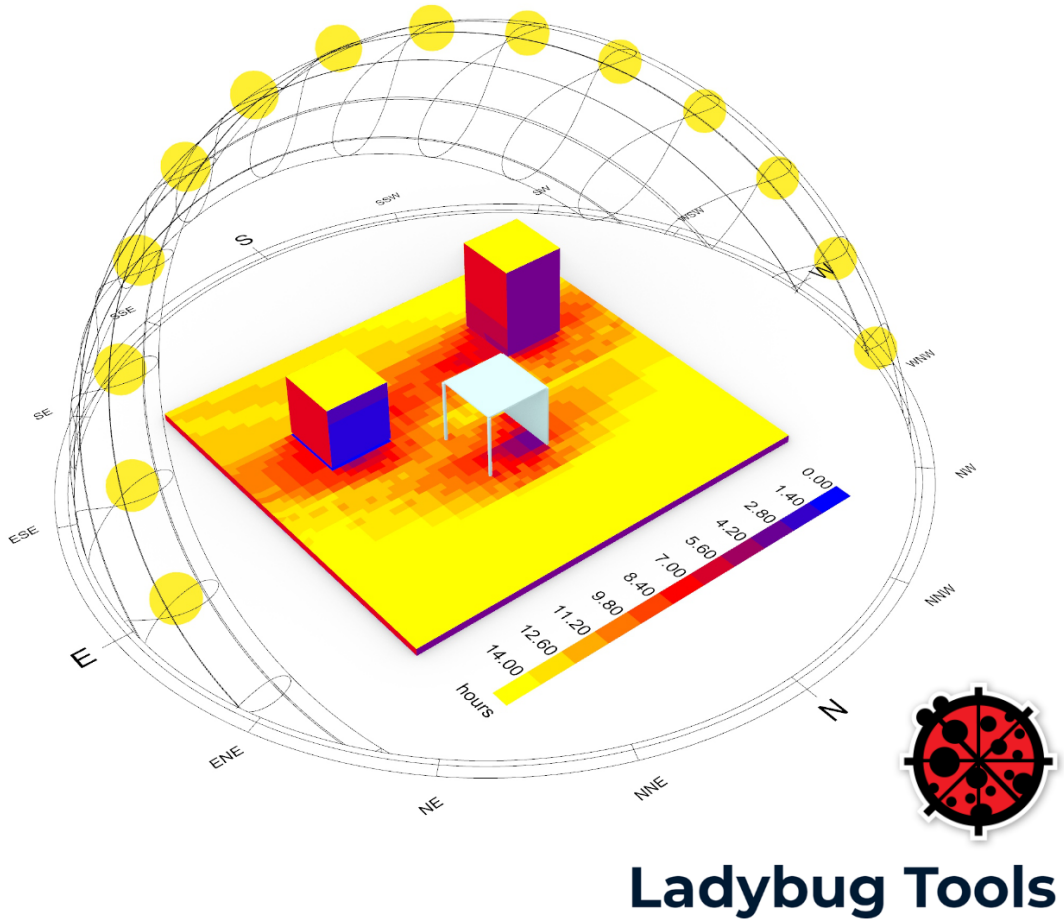
Práce s komponenty Karamba3D je velmi intuitivní a pro uživatele tradičních programů založených na MKP i velmi logická a jednoduchá. Vytvoření modelu se skládá z definování prutových nebo plošných prvků, určení podpor, zatížení nebo volby materiálu. Dále jsou na výběr řešiče prvního řádu, druhého řádu, stabilitní analýza, analýza teorie velkých deformací aj. Zobrazení výsledků a případná práce s nimi je také velmi intuitivní. Karamba3D má navíc velmi dobře zpracovaný manuál, který je dostupný na jejich webových stránkách. Práce s balíčkem je pak více popsána v kapitole 5.

2.1.3 Ladybug Tools

Dalším rozšířením používaným v diplomové práci je sada nástrojů Ladybug Tools. Jedná se o aplikace, které spojuje téma environmentální analýzy čili analýzy prostředí. Pomocí Ladybug Tools lze pak jednoduše získat data o klimatu a aplikovat je na navrhovanou stavbu. Sada se skládá z několika hlavních částí, konkrétně z: Ladybug, Honeybee, Butterfly a Dragonfly.

Pomocí první aplikace Ladybug lze získat a zobrazit grafy klimatických dat, analyzovat pohyb slunce po obloze a z toho vyvodit oslunění, proslunění či zastínění budov. Dále lze např. analyzovat výhledy z interiéru do exteriéru nebo posuzovat míru vnějšího tepelného komfortu. Pomocí Honeybee lze provádět detailnější analýzy tepelné techniky, denního osvětlení aj. Plugin Butterfly je pak zaměřen na CFD analýzy proudění, a tak je určena především na simulaci větru jak ve vnějším, tak i ve vnitřním prostředí. Balíček Dragonfly je zaměřen na spotřebu energií v rámci několika budov nebo i městských částí, a také na optimalizaci využívání obnovitelných zdrojů elektřiny.

Rozšíření Ladybug Tools spojuje všechny programy používaných při analýzách prostředí a vytváří tak jednotný balíček přidružených programů, pomocí kterého lze navrhnout efektivní stavbu s ohledem na vlivy prostředí.



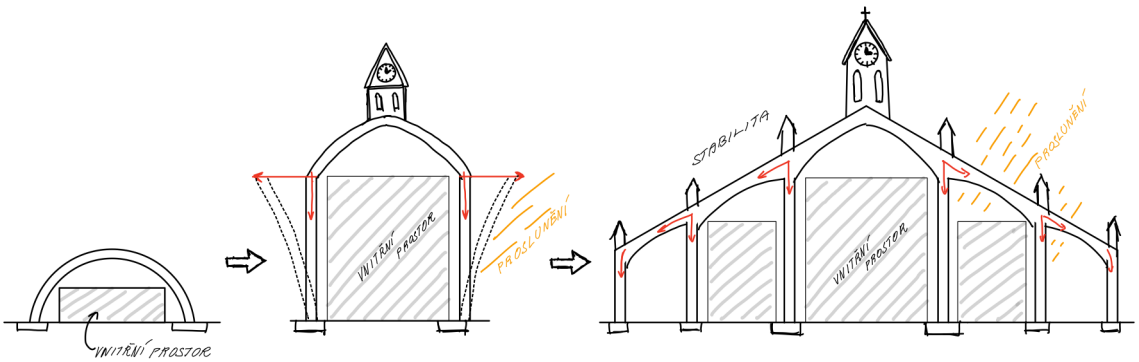
Ladybug Tools

Obrázek 2.5: Ukázka použití softwaru Ladybug

Kapitola 3

Optimalizace konstrukce

Obecně je optimalizace proces, během kterého hledáme nejvhodnějšího řešení určitého problému. Nejvhodnější nebo nejlepší řešení je pak označováno jako optimum či optimální řešení. Například optimalizaci návrhu stavební konstrukce můžeme provádět dle předem určených kritérií, dle kterých budeme daný návrh hodnotit. Příkladem jednoduché a představitelné optimalizace může být dřívější využívání závěsných modelů popsané v kapitole 2. Cílem těchto závěsných modelů bylo optimalizovat velké světské stavby, jako kostely či chrámy, tak aby tvar oblouků a kleneb byl co nejvíce staticky efektivní, a tak bylo nutné maximalizovat tlakové síly a minimalizovat ohybové momenty viz princip řetězovky. Pokud by ale statika byla jediným kritériem optimalizace, tak by se nám spíše zachovaly stavby podobné kryptám či iglú. Pro tehdejší světské stavby byla důležitá velkolepost a monumentálnost. Má to své důvody v otázkách víry a náboženství, ale velikost těchto prostor měla i praktická opodstatnění, například bylo nutné vytvořit velký prostor pro dostatečný počet lidí nebo přivést co nejvíce přirozeného světla do vnitřních prostor apod. Každopádně to pro architekta či stavitele nebyla jednoduchá úloha a kritérií, které bylo nutné splnit, bylo určitě více.



Obrázek 3.1: Příklad vývoje konstrukce

Generativní navrhování

V dnešní době spolu s parametrickým modelováním vzniká fenomén tzv. generativního navrhování (designu), který je velmi populární hlavně ve strojírenství. Pomocí této metody se navrhují například součástky v automobilovém nebo leteckém průmyslu. V architektuře se tímto způsobem mohou generativně navrhovat vnitřní prostory, dispozice nebo vnější tvary budovy. V pozemním stavitelství to pak například mohou být jednotlivé konstrukční prvky. Generativní navrhování je metoda, která vzniká spojením parametrického modelování a optimalizačního procesu. Uživatel tak stejně jako u parametrického modelování nezadává cílovou geometrii, ale pouze oblast geometrických parametrů a ostatní vstupní údaje. Program následně vygeneruje např. sadu optimalizovaných tvarů dle zvolených kritérií, z kterých uživatel může vybrat vhodné řešení.

3.1 Optimalizace pomocí evolučních algoritmů

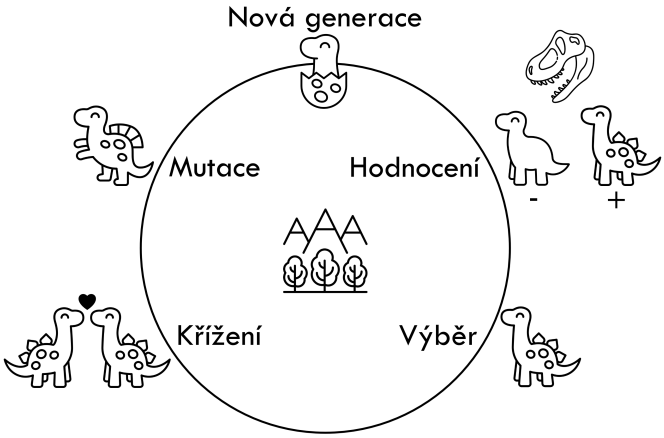
K řešení optimalizačních úloh byly v diplomové práci využívány genetické algoritmy. K efektivnímu využívání těchto algoritmů je vhodné znát princip jejich fungování, proto v této sekci bude popsán obecný princip evolučních algoritmů a představen bude také využívaný program včetně konkrétního genetického algoritmu.

Obecně se dá evoluční algoritmus popsat jako algoritmus využívaný k řešení optimalizační úlohy, který je založen na principech evoluční biologie a přirozeného výběru, jehož podmínky definoval Charles Darwin již v polovině 19. století. Přirozený výběr spočívá ve zvýhodňování těch jedinců, kteří splňují určitá kritéria, a naopak potlačuje jedince, kteří kritéria nesplňují. Mezi základní principy patří možnost křížení, mutace či selekce(výběru).

Genetický algoritmus

Genetický algoritmus spadá mezi evoluční algoritmy a je velmi často využíván jako nástroj optimalizace ve zmíněném generativním navrhování. Principem je využít pravidla evoluce k nalezení nejvhodnějšího jedince napříč generacemi. Velikost a počet generací je pak možnou volbou v nastavení algoritmu. Vhodnost jedince se posuzuje podle kritérií, které se označují jako fitness funkce a každý jedinec podle ní nabývá určité hodnoty fitness neboli zdatnosti či kvality. Jedinci jsou v algoritmu charakterizováni informacemi, které se nazývají geny. Soubor všech genů se pak označuje jako genom, který se konkrétně v generativním navrhování skládá z

jednotlivých proměnných parametrů. Princip přirozeného výběru pak v genetickém algoritmu spočívá v předávání vybraných genů dalším generacím, což ovlivňují jednotlivé fáze algoritmu jako fáze ohodnocování jedinců, fáze selekce (výběru), reprodukce (křížení) a mutace. Tyto fáze budou podrobněji rozebrány níže v textu.

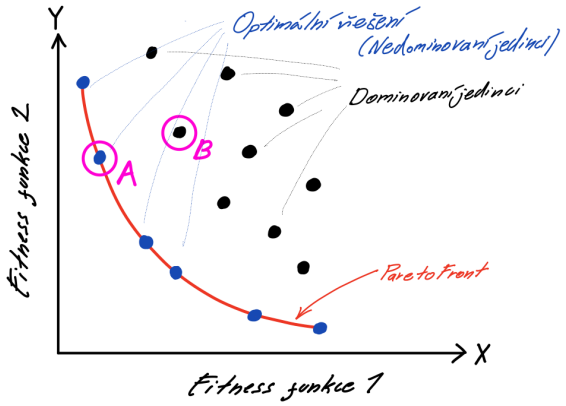


Obrázek 3.2: Znázornění fází genetického algoritmu

Na Obrázku 3.2 lze vidět jednotlivé fáze algoritmu v pořadí, ve kterém genetický algoritmus postupuje. Na začátku procesu je nutné stanovit první generaci, která se určí zadáním velikosti populace (počet jedinců v generaci) a většinou náhodnou volbou genů.

Hodnocení

Následuje fáze ohodnocení, ve které dojde k porovnání jedinců s fitness funkcí. Na základě toho je každému jedinci přiřazena hodnota zdatnosti (fitness). Hodnota může nabývat například od 0 do 1, s tím, že chtěný výsledek je např. co nejmenší (minimalizovat fitness).



Obrázek 3.3: Znázornění grafu se zvýrazněnou Pareto frontou

Jedinec může být také algoritmem porovnáván podle více kritérií, potom se mluví o vícekriteriální optimalizaci. Jedním z často používaných vyhodnocení vícekriteriální optimalizace je tzv. **Pareto front** (Paretovo optimum), což je určení nejvhodnějších jedinců, které podléhají dvou a více kritériím. Na Obrázku 3.3 je schéma grafu, na kterém je vyznačeno Paretovo optimum. Skládá se z jedinců, kteří nebyli dominováni žádným ostatním jedincem, a proto splňují podmínku a mohou být označení za optimální řešení. Podmínka dominance pro minimalizaci kritérií bude vysvětlena na jedincích A a B označených na Obrázku 3.3. Jedinec $A(x_1, y_1)$ je dominantní vůči jedinci $B(x_2, y_2)$, pokud $(x_1 \leq x_2 \wedge y_1 \leq y_2) \wedge (x_1 < x_2 \vee y_1 < y_2)$.

Selekce - výběr

Podle hodnocení jsou jedinci podrobeni výběru ke křížení, aby se jejich geny mohly přenést do další generace ve formě nového jedince. Je několik metod selekce, které budou popsány v následujících odstavcích, ale obecně platí, že jedinci s lepší fitness hodnotou jsou upřednostňováni (viz přirozený výběr).

Ruletová selekce spočívá ve výběru jedinců podle pravděpodobnosti, která je přiřazena každému jedinci podle jeho fitness hodnoty f . Matematicky lze pravděpodobnost $p(i)$ vyjádřit následovně:

$$p(i) = \frac{f(i)}{\sum_{j=1}^{n} f(j)}$$

Nevýhoda této metody je riziko předčasné konvergence k lokálnímu optimum, ke které dochází, pokud se v generaci objeví dominantní jedinec, který neustále vyhrává ruletovou selekci [6].

Turnajová selekce je metoda výběru, při které se jedinci náhodně rozřadí do skupin. Skupina sestává z minimálně dvou jedinců a cílem selekce je vybrat nejvhodnějšího čili nejlépe hodnoceného jedince ze skupiny. Tento jedinec tímto vyhrává pomyslný turnaj a je vybrán ke křížení. [6]

Reprodukce - křížení

Následují dvě fáze tzv. genetických operátorů, kde prvním z nich je křížení jedinců. Vybraní jedinci z fáze selekce jsou nyní určeni ke stvoření svého potomka, kterému předají své genetické informace. Proces reprodukce může být ovlivněn pravděpodobností křížení, která bývá většinou vysoká, a udává s jakou pravděpodobností dojde k výměně genů mezi rodiči. Pokud by byla šance na zkřížení malá, tak by větší část potomků byla kopiemi rodičů.

Mutace

Druhým z genetických operátoru je mutace, která vnáší určitou náhodnost do procesu evoluce. Mutace je v genetickém algoritmu s malou pravděpodobností aplikována na potomky vzniklé po zkřížení. Úkolem mutace je náhodně pozměnit část zděděných genů a odlišit potomky od svých rodičů. Tím se také do určité míry zajistí genetická diverzita neboli různorodost, která je v přírodě důležitá pro vývoj a adaptaci na nové prostředí. Mutace má další přínos ve formě fungování genetického algoritmu, ve kterém snižuje riziko nalezení lokálního optima.

3.1.1 NSGA-II

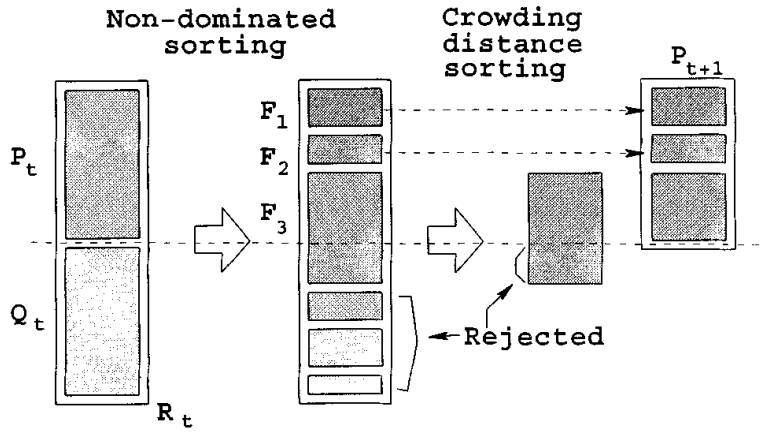
Z obecného principu popsaného výše se dostáváme ke konkrétnímu genetickému algoritmu, který jako jeden z mnoha vylepšuje a zavádí nové funkce do obecného fungování genetického algoritmu. NSGA-II je zkratka pro Non-dominated sorting genetic algorithm II, což je genetický algoritmus pro vícekriteriální optimalizaci, kterého je využíváno v optimalizační úloze diplomové práce. Algoritmus má tři klíčové vlastnosti, a to: [7]

1. *Elitismus* - nejlepší jedinci populace se zachovávají do dalších generací
2. *Crowding distance* - zachování rozmanitosti, tzn. třídění podle hustoty jedinců okolo každého jedince, ti s menší hustotou jsou preferováni
3. *Zvýhodnění nedominovaných řešení*

Průběh algoritmu NSGA-II je znázorněn na Obrázku 3.4 a na následujících řádcích bude vysvětlen. Nejprve bude vysvětlena inicializační fáze, a v dalším odstavci bude popsána fáze běžná. Inicializační fáze algoritmus začíná náhodným zvolením první generace označované P_0 o velikosti N, která je ohodnocena dle fitness funkcí a seřazena od nedominovaných (nejlepších) jedinců po nejvíce dominované (nejhorší) jedince. Následně jsou jedinci vybíráni ke křížení pomocí turnajové selekce. Vybraní jedinci, nyní už rodiče, jsou určeny ke křížení a po aplikování mutace dají vzniknout potomstvu označovaném Q_0 o velikosti N. [8]

V běžné fázi je pak prvně vytvořena kombinovaná generace R_t, která se skládá z P_t rodičů a Q_t potomků. Tato kombinovaná generace o velikosti $2N$ zaručuje princip *Elitismu*, pomocí kterého se dokážou zachovat nejlepší jedinci i z předešlé generace. Následuje ohodnocení a řazení dle dominance do skupin, kde první skupina označovaná F_1 je skupina nedominovaných (nejlepších) jedinců. K tvorbě další

generace je však nutné vybrat pouze N jedinců / rodičů, proto je kombinovaná generace oříznuta o nejhůře hodnocené jedince. V tomto řazení mají velký význam zmiňované skupiny, protože prvně je vybrána skupina F_1, a následně pokud $F_1 < N$, tak je množina N jedinců doplněna ostatními skupinami. Pokud nastane situace, že se do výběru nevejde celá skupina, tak je využito tzv. *Crowding distance*, kde přednost dostanou jedinci s nižší hustotou sousedících jedinců. Tím vznikne generace rodičů P_{t+1}, která opět podstoupí selekci a následně křížením a aplikací mutace dá vzniknout novému potomstvu Q_{t+1} o velikosti N. [8]



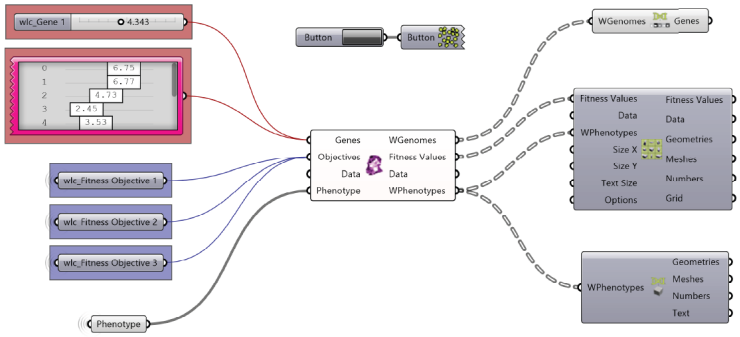
Obrázek 3.4: Proces fungování NSGA-II [8]

3.1.2 Wallacei

Wallacei je rozšiřující balíček do prostředí Grasshopper, který slouží k evolučním simulacím. Balíček je pojmenován po britském přírodopisci, badateli a biologovi, který se jmenoval Alfred Russell Wallace a nezávisle na Darwinovi popsal evoluční princip přirozeného výběru. [9]

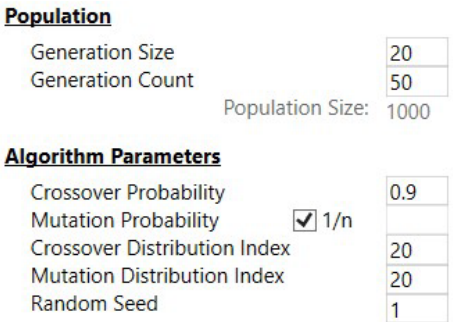
V diplomové práci je balíček používán jako hlavní optimalizační nástroj pro řešení problému optimalizace sloupových prvků. Wallacei se skládá z dvou částí, a to Wallacei X a Wallacei Analytics. Wallacei X je evoluční řešič, který je založen na algoritmu NSGA-II viz předchozí sekce 3.1.1, pomocí kterého jsou prováděný veškeré evoluční simulace, a je tedy hlavní částí balíčku. Wallacei Analytics slouží k následné analýze a grafickému zobrazení výsledků v prostředí Rhino.

Na Obrázku 3.5 je příklad zapojení Wallacei v prostředí Grasshopper, který se skládá z hlavního řešiče Wallacei X uprostřed a komponent Wallacei Analytics napravo na obrázku. Nalevo jsou vstupní data do genetického algoritmu, které sestávají z genů ve formě proměnných hodnot, fitness kritérií a fenotypu, který obsahuje většinou geometrické tvary, které můžou být závislé na genech a slouží k následnému zobrazení optimalizovaných výsledků.



Obrázek 3.5: Zapojení Wallacei v prostředí Grasshopper [9]

Po dvojkliku na komponentu Wallacei X je možné nastavit parametry evoluční simulace. Základní nastavení je velikost a počet generací, z čehož vyplývá celková velikost populace. Níže je nastavení parametrů algoritmu, kde se dá ovlivnit pravděpodobnost křížení *(Crossover Probability)* vysvětlena zde 3.1 nebo pravděpodobnost mutace *(Mutation Probability)* popsána zde 3.1, která je doporučena velmi malá, např. výchozí nastavení je $1/n$, kde n je počet proměnných (No. of Values). Další možností je nastavení *Crossover* a *Mutation Distribution Index*, kterými se dá ovlivnit rozmanitost populace. Indexy nabývají hodnot od 0 do 100 s tím, že vyšší hodnoty znamenají větší pravděpodobnost, že potomek nebo provedená mutace bude více podobná neboli málo vzdálená rodičům. Naopak indexy bližší k 0 vytváří rozmanitější generace, jelikož umožňují výběr vzdálených řešení jako potomků. Posledním nastavením je *Random Seed*, kterým se dá ovlivnit inicializační fáze, a udává, zda algoritmus bude začínat vždy od jiné sady náhodných čísel/genů pokud zvolíme hodnotu 0, nebo vždy od stejné sady pokud zvolíme jakékoliv číslo větší od 0. [9]

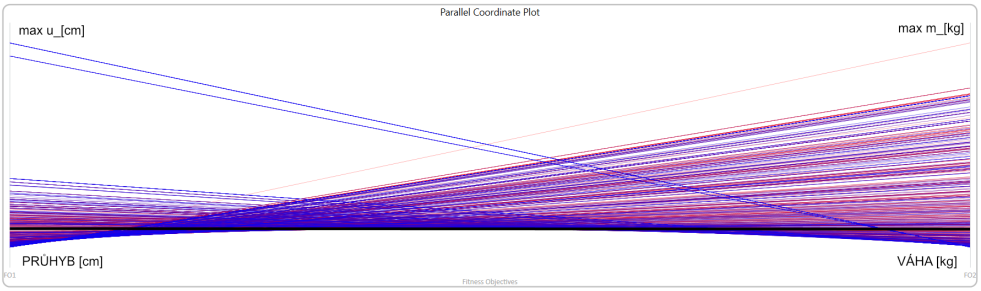


Obrázek 3.6: Nastavení algoritmu Wallacei X

Před spuštěním algoritmu si uživatel může navolit živé zobrazování výsledků na grafu PCP (Parallel Coordinate Plot) nebo na grafu směrodatné odchylky či na 3D grafu kritérií. Následně lze ještě zapnout funkci *Autosave*, která ukládá data po

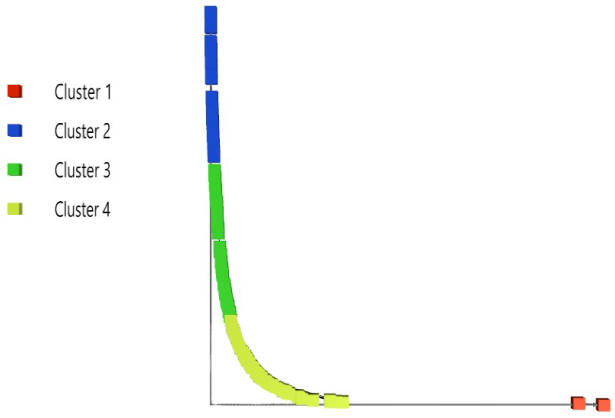
každé generaci pro případ neočekávaného ukončení programu. Poslední možnost je před zapnutím minimalizovat okno Rhino, což velice urychluje čas výpočtu z důvodu nezobrazování optimalizované geometrie během simulace. Po zapnutí lze vidět pod záložkou **RunTime** očekávaný čas výpočtu a další informace o stavu.

Po dokončení simulace může uživatel analyzovat výsledky na následující záložce v okně Wallacei X, která se jmenuje Wallacei Analytics. Zde lze zobrazit výsledky na grafech směrodatných odchylek pro všechna kritéria, na grafech průměrné hodnoty fitness, na paprskovém grafu a dalších.



Obrázek 3.7: Příklad PCP grafu s kritérii průhybu a váhy konstrukce

Hodnoty pro export může uživatel zvolit na následující záložce Wallacei Selection. Výsledky lze vybrat například z PCP grafu, který ohodnocené výsledky přiřazuje na vertikální osu od nejlepších (nejmenších) jedinců po nejhorší viz Obrázek 3.7. PCP graf pak lze vyhodnotit pomocí čtyř metod analýzy grafu. Dále lze například zobrazit Pareto Front řešení, která je vysvětlena zde 3.1. Uživatel může také vybrat řešení pomocí shlukové analýzy, která využívá K-Means algoritmu, a rozděluje řešení do tzv. shluků(clusterů) neboli skupin, které tvoří jedinci vzdálenostně podobní. Uplatnění shlukové analýzy na 2D Pareto front je znázorněno na Obrázku 3.8, kde jsou shluky nazývány anglickým názvem *Cluster*.

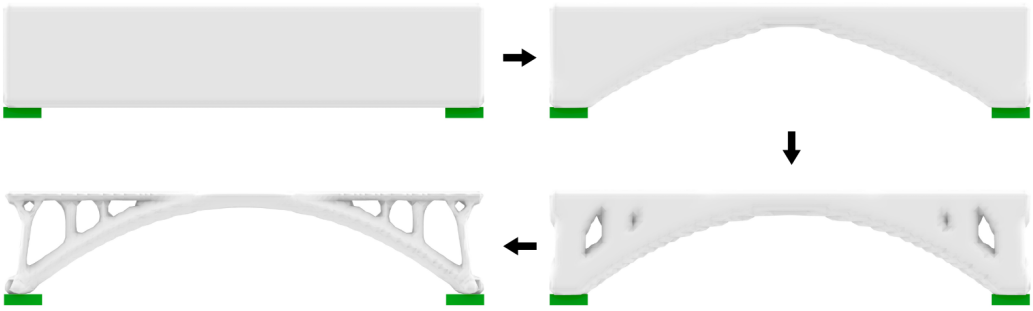


Obrázek 3.8: Shluková analýza aplikována na Pareto Front

3.2 Topologická optimalizace

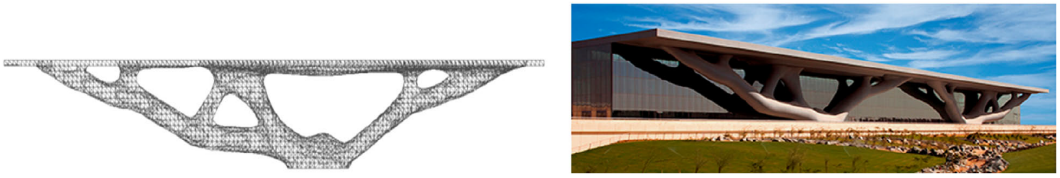
V předešlé sekci 3.1 byly popsány základní principy fungování genetických algoritmů, a také konkrétní algoritmus využívaný v diplomové práci. Samotný genetický algoritmus však neslouží jako hotový program pro optimalizaci konstrukce, nýbrž pouze jako nástroj, okolo kterého je nutné vybudovat prostředí k zadávání vhodných vstupních údajů a k analýze výstupních údajů.

V této sekci bude opět využíváno rozšiřujících balíčku do GH, ale již rozšíření předurčených k optimalizaci konstrukce nebo obecněji k optimalizaci tvaru. Optimalizace tvaru neboli tzv. topologická optimalizace je velmi rozšířená zejména ve strojírenství a slouží např. k výrobě odlehčených, ale zároveň pevných dílů. Vychází z pojmu topologie, v které na rozdíl od geometrie nezáleží na vzdálenostech, křivostech a dalších geometrických vlastnostech, ale záleží na způsobu, jak jsou tvary mezi sebou propojeny či jak spolu sousedí [10]. Z hlediska topologie jsou tvary jako krychle, kvádr, jehlan nebo koule rovnocenné, resp. v názvosloví topologie jsou vzájemně homeomorfní [10].



Obrázek 3.9: Příklad TO spojitě zatíženého nosníku provedené metodou SIMP

Topologická optimalizace je forma strukturální optimalizace, u které je cílem optimální rozložení materiálu uvnitř navrhovaného prostoru. Hledané optimum je závislé na metodě TO a na okrajových podmínkách. V architektuře a pozemním stavitelství se tato metoda nevyužívá tak často jako ve strojním odvětví, ale jsou již známé hotové projekty, kde bylo principu TO využito, jako např. na Obrázku 3.10 při návrhu výstaviště v Kataru. Hlavním úskalím TO je zejména pak proveditelnost návrhu, který je vetšinou organického nepravidelného tvaru. Ve strojírenství je tento problém řešen moderním způsobem, a to využitím například 3D tisku nebo CNC obrábění. Faktem je, že se tato metoda využívá na díly, které jsou velikostně několikanásobně menší než stavební konstrukce. Metoda 3D tisku v architektuře a pozemním stavitelství už ale také není minulostí. Dnes se využívá např. přímé 3D tisknutí betonu nebo výroby bednění pomocí 3D tisku.

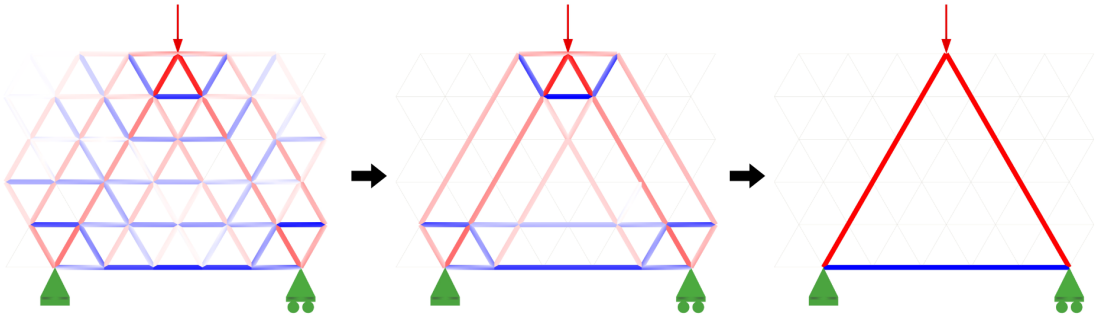


Obrázek 3.10: Qatar National Convention Centre, upraveno podle [11]

Z tohoto důvodu budou v této sekci uvedeny také ostatní příklady TO zaměřené na optimalizaci sloupových prvků. TO konstrukcí lze rozdělit na *optimalizaci dispozice* (Layout Optimization - LO) a na *zobecněnou optimalizaci tvaru* (Generalized Shape Optimization - GSO). Optimalizace dispozice je zaměřena na vhodné rozložení prutových prvků a určena pro konstrukce, které mají malý objem ve srovnání s návrhovou oblastí. Naopak u zobecněné optimalizace tvaru, jak už z názvu vyplývá, jde spíše o návrh tvaru se zachováním topologických pravidel. Optimalizované tvary mají také větší objemový podíl na rozdíl od konstrukcí v LO. [12]

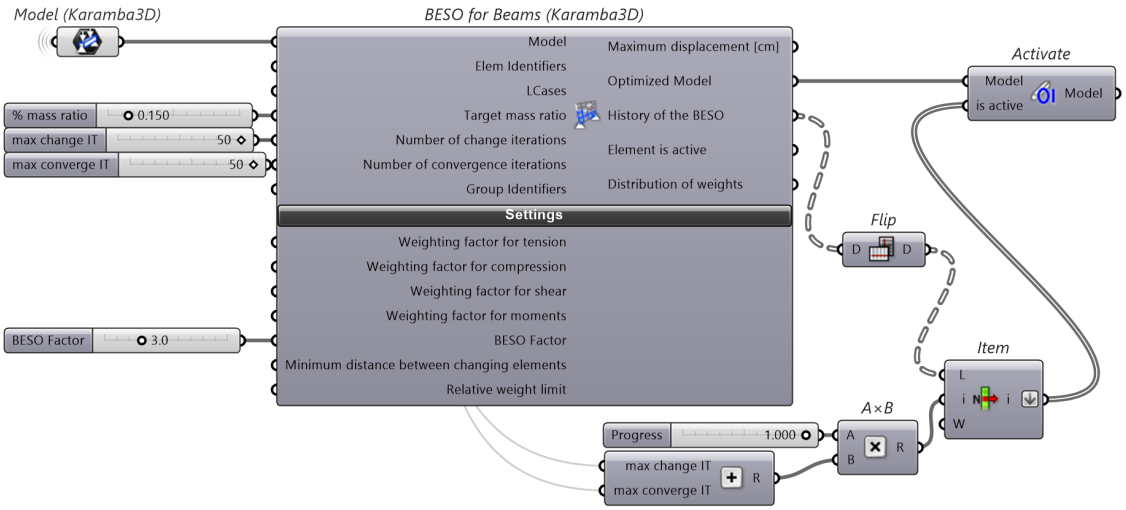
3.2.1 Optimalizace dispozice (LO) s využitím BESO

K *optimalizaci dispozice* bylo využito rozšíření Karamba3D popsaného v sekci 2.1, konkrétně komponenty *BESO for Beams* (určená pouze pro prutové prvky), která využívá optimalizační metody BESO (Bi-Directional Evolutionary Structural Optimization). BESO je iterativní metoda topologické optimalizace, která je založena na MKP a ESO (Evolutionary structural optimization), což je starší metoda TO, která spočívá v postupném odebírání staticky neefektivního materiálu, a tím vytvoření optimálního tvaru konstrukce. Metoda BESO na rozdíl od ESO umožňuje během iterací zároveň jak odebírání, tak přidávání materiálu do míst, kde je potřebný. Samotná metoda BESO je využitelná i pro zobecněnou optimalizaci tvaru, např. v komerčním rozšíření pro GH Ameba nebo v samostatných programech jako Ansys, Autodesk Fusion aj.



Obrázek 3.11: Příklad optimalizace rámu pomocí *BESO for Beams*

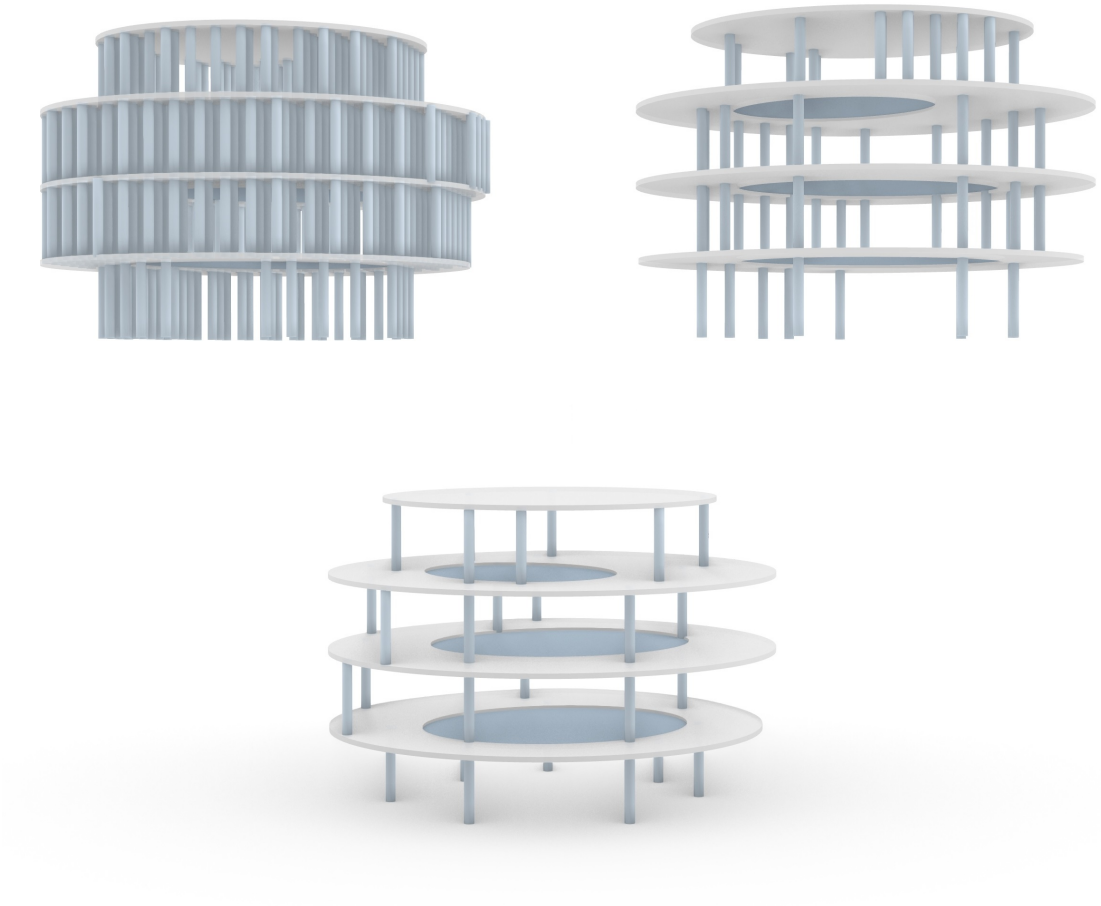
Schéma provedené optimalizace pro seznámení s komponentou lze vidět na Obrázku 3.11. Pro vytvoření jednoduchého příkladu byla vytvořena trojúhelníková síť, která byla zaplněna prutovými prvky. Následně jsou zadána základní data pro analýzu konstrukce pomocí Karamba3D. Konstrukce je podepřena pevným kloubem nalevo, posuvným kloubem na pravé straně a uprostřed zatížena. Sestavený model lze následně zapojit do optimalizační komponenty *BESO for Beams*.



Obrázek 3.12: Zapojení *BESO for Beams* v prostředí Grasshopper

Na Obrázku 3.12 je vidět použité nastavení algoritmu pro příklad rámu. Základní hodnota je *Target mass ratio*, která udává poměr cílové hmoty konstrukce k hmotě počáteční. V příkladu je zvolena 15% hmota, tedy požadovaný výsledek je odebrání 85 % hmoty konstrukce. Dále se nastavuje počet iterací, v kterých má algoritmus dosáhnout požadované hmoty *(Change iterations)* a počet konvergenčních iterací *(Convergence iterations)*, během kterých se již podíl hmoty nemění, ale algoritmus může vylepšovat polohu prutů. V nastaveních ve spodní části komponenty lze nastavit váhové faktory silám/momentu nebo upravit *BESO Factor*, kterým lze upravit obousměrnost *(bi-directionality)* algoritmu. Faktorem se násobí počet prvků, které se mají v každé iteraci odebrat a přidat.

Hlavním výstupem z komponenty je optimalizovaný model, se kterým se pracuje stejně jako s běžným modelem v prostředí Karamba3D (Analyze → Model View ...). Dalším využívaným výstupem je historie algoritmu *(History of the BESO)*, pomocí jenž lze zobrazit proběhlé iterace algoritmu. K zobrazení historie algoritmu na modelu může uživatel využít např. sestavení komponent zobrazené na Obrázku 3.12.



Obrázek 3.13: Využití *BESO for Beams* na optimalizaci polohy sloupů

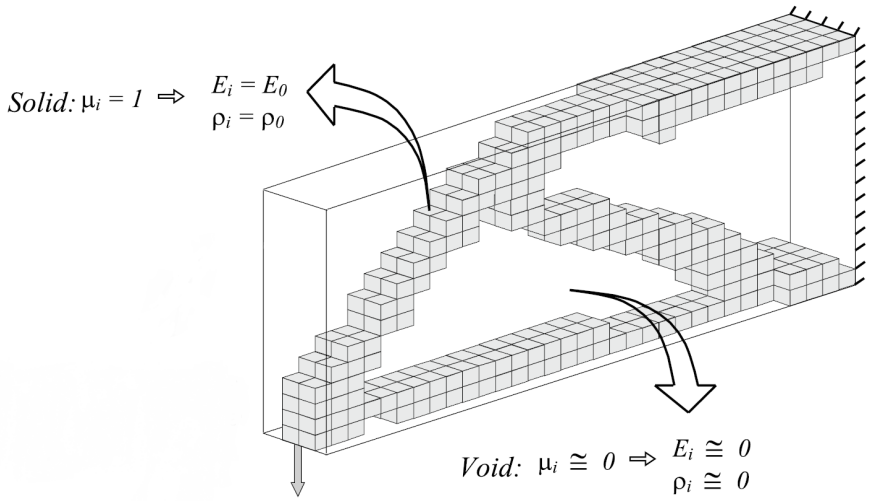
Algoritmus *BESO for Beams* byl následně testován na úloze optimalizace polohy sloupů. Kvůli tomu byl namodelován nepravidelný tvar budovy pomocí parametrického modelování, v němž byly navrženy zóny, v kterých ideálně nemají být žádné sloupy. V praxi mohou být tyto prostory definovány architektem, např. jako zasedací místnosti, nebo átria či jiné volné prostory. Pro využití BESO algoritmu byl návrhový prostor hustě zaplněn sloupy, které může algoritmus odstraňovat a přidávat k nalezení vhodného rozložení. Dále byly zadány vstupní data pro statickou analýzu ve formě plošného zatížení od desek a definování podpor. BESO algoritmus byl nastaven na požadovaných 5 % hmoty. Na Obrázku 3.13 pak lze vidět proces algoritmu, kde model nejníže je výstupní optimalizovaný model.

Proces optimalizace pomocí *BESO for Beams* je poměrné rychlý, i když samozřejmě záleží na počtu iterací a velikosti úlohy, a k řešení algoritmus konvergoval též rychle. Ovšem algoritmus často konvergoval k nesmyslným řešení a až po dlouhém nastavování počtu iterací a hodnoty *BESO Factor* se podařilo dojít k smysluplnější

optimalizaci, která je právě ukázána na Obrázku 3.13. Nejčastějším problémem bylo, že algoritmus se snažil umístit potřebnou hmotu do určitých míst tím způsobem, že nashromáždil sloupy na jedno místo a zbytek konstrukce byl nepodepřen. Bylo znát, že komponenta *BESO for Beams* je určená spíše pro příhradové nebo roštové konstrukce a využití na řešený problém není ideální.

3.2.2 Optimalizace tvaru (GSO) s využitím metody SIMP

Pro *zobecněnou optimalizaci tvaru* bude využíváno optimalizační metody SIMP ve formě pluginu pro GH *tOpos* [13]. Metoda SIMP (Solid Isotropic Material with Penalization) je společně s BESO jedna z nejčastěji používaných metod pro topologickou optimalizaci. Metoda SIMP stejně jako BESO vychází ze sítě konečných prvků, kde je každý prvek vyplněn materiálem, resp. v každém prvku je určitá relativní hustota materiálu ρ, která nabývá hodnot od 0 do 1. Hodnoty vyšší, tedy bližší číslu 1 značí, že je prvek naplněn materiálem, zatímco prvky s hodnotou 0 neobsahují materiál. Metoda BESO dovoluje pouze hustotu $\rho = 1$, nebo $\rho = 0$, tedy neumožňuje hodnoty mezilehlé, zatímco metoda SIMP ano.

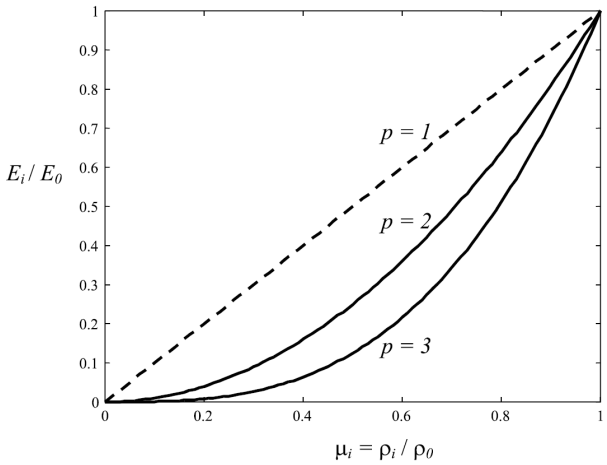


Obrázek 3.14: Optimalizace konzole pomocí SIMP a znázornění hustoty ρ, upraveno podle [14]

Na Obrázku 3.14 lze vidět plné prvky (kostičky) s hustotou $\rho = 1$ a prázdné prvky v návrhovém prostoru konzole s $\rho \approx 0$. Jelikož se relativní hustota ρ_i může v každém i-tém prvku měnit, tak se mění i modul pružnosti E v každém i-tém prvku dle vztahu:

$$E(\rho_i) \;=\; \rho_i^p \; E_0$$

E_0 v rovnici vyjadřuje základní nebo počáteční modul pružnosti pro materiál, kterého nabývá pokud $\rho_i = 1$ a písmenem p v exponentu se značí tzv. penalizační součinitel. Součinitel p má za úkol zmenšovat význam mezilehlých prvků tím, že zhoršuje relativní modul pružnosti v závislosti na relativní hustotě. Součinitel penalizace je znázorněn na Obrázku 3.15, kde lze vidět různé křivky pro určité hodnoty penalizačních faktorů. Obecně se udává, že hodnota $p = 3$ vykazuje nejlepší výsledky a je doporučena. V pluginu *tOpos* se využívá hodnot 2 až 3. Z grafu lze pak vyčíst, že s penalizačními faktory $p > 1$ jsou upřednostňovány hodnoty hustoty ρ blízké 0 a 1.



Obrázek 3.15: SIMP - Graf ukazující význam penalizačního součinitele p [14]

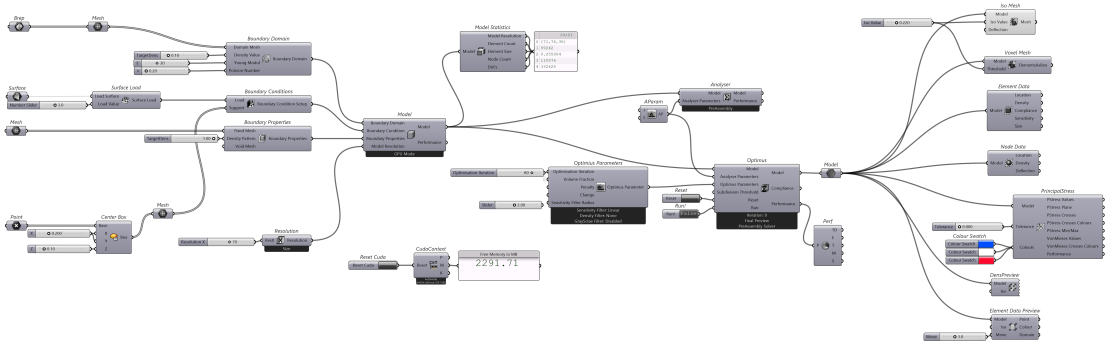
Jak už bylo zmíněno, pro optimalizaci tvaru bylo využíváno rozšiřujícího balíčku *tOpos*, který vytvořil Sebastian Białkowski [13] jako nástroj pro architekty s využitím v rané fázi návrhu. Cílem autora bylo vytvořit jednoduchý, dostupný software, který by umožňoval rychlé návrhy pomocí topologické optimalizace. Dostupnost a rychlost byly také hlavní důvody pro použití pluginu *tOpos* v diplomové práci.

Software využívá algoritmu SIMP s funkcí maximalizace tuhosti současně s odebíráním hmoty konstrukce. Přesněji uvedeno, algoritmus minimalizuje globální poddajnost C, což je ekvivalentní proces ke globální maximalizaci tuhosti tak, že celková hustota prvků má být v každém kroku odstraňovaní hmoty co nejlépe konstrukčně využita. Matematicky lze základní principy optimalizačního algoritmu vyjádřit takto [15]:

$$min\ C[\{\rho\}] \ = \ U^T \ K \ U \ = \ \sum_{i=1}^{N} (\rho_i)^p \ [u_i]^T \ [K_i] \ [u_i]$$

$$[K\{\rho\}] \ \{u\} \ = \ \{F\}$$

Při vytváření modelu v rozhraní pluginu *tOpos* musí uživatel vytvořit menší definici pomocí několika komponent, které balíček nabízí. Nevýhoda pluginu je, že postrádá návod, který by jednotlivé komponenty vysvětloval, a který by uživatele provedl fungováním balíčku. K dispozici jsou jen vzorové soubory, podle kterých lze sestavit komponenty k řešení vlastního problému.

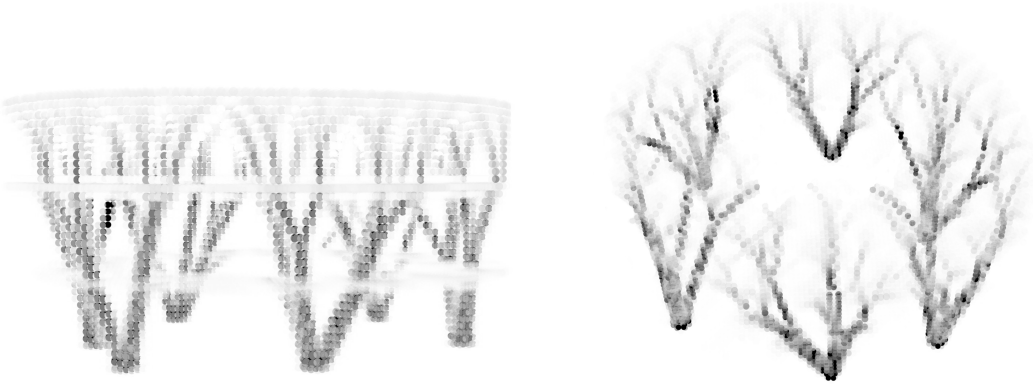


Obrázek 3.16: Zapojení pluginu *tOpos* v prostředí Grasshopper

Vstupními údaji při optimalizaci sloupových prvků byly: celý návrhový objem konstrukce, místa zatížení konstrukce a místa podpor. Pro algoritmus bylo nastaveno 80 iterací s penalizačním faktorem mezi 2 až 3. Požadovaná hmota konstrukce na Obrázku 3.17 byla 20 %.

Obrázek 3.17: Pohled na konstrukci po optimalizaci v pluginu *tOpos*

Tvar výsledné konstrukce je velmi organický, což je u optimalizace tvaru běžné, jelikož si síly v konstrukci hledají nejoptimálnější cestu do míst podpor a nejsou nikterak omezovány. Vzniklou konstrukci tak tvoří větvené sloupy, které jsou ve tvaru stromů. Metoda se využívá v rané fázi návrhu a zejména při hledání vhodného tvaru konstrukce (form finding), kde metoda může nabídnout opravdu přínosné a různorodé varianty konstrukce. Návrh by v této formě bylo vhodné ještě upravit a zamyslet se nad využitím 3D tisku, který má u podobných konstrukci velký potenciál.



(a) Pohled zepředu (b) Pohled shora

Obrázek 3.18: Zobrazení konstrukce z hlediska hustoty prvků ρ

Topologická optimalizace je určitě metoda, která má v navrhovaní stavebních konstrukcí budoucnost a již se využívá zejména z architektonického hlediska. Velký potenciál má však i z hlediska statického, jelikož dokáže eliminovat zbytečný materiál v konstrukcích a ušetřit tak materiál i finance při výrobě. Problémem však zůstává zmíněná proveditelnost zejména větších konstrukcí.

Jelikož je v diplomové práci záměrem, aby optimalizovaná konstrukce byla i proveditelná a výstavbu zbytečně nekomplikovala, nebude již metoda topologické optimalizace využívána a místo toho bude implementován vlastní algoritmus na řešení optimalizace sloupů.

Kapitola 4

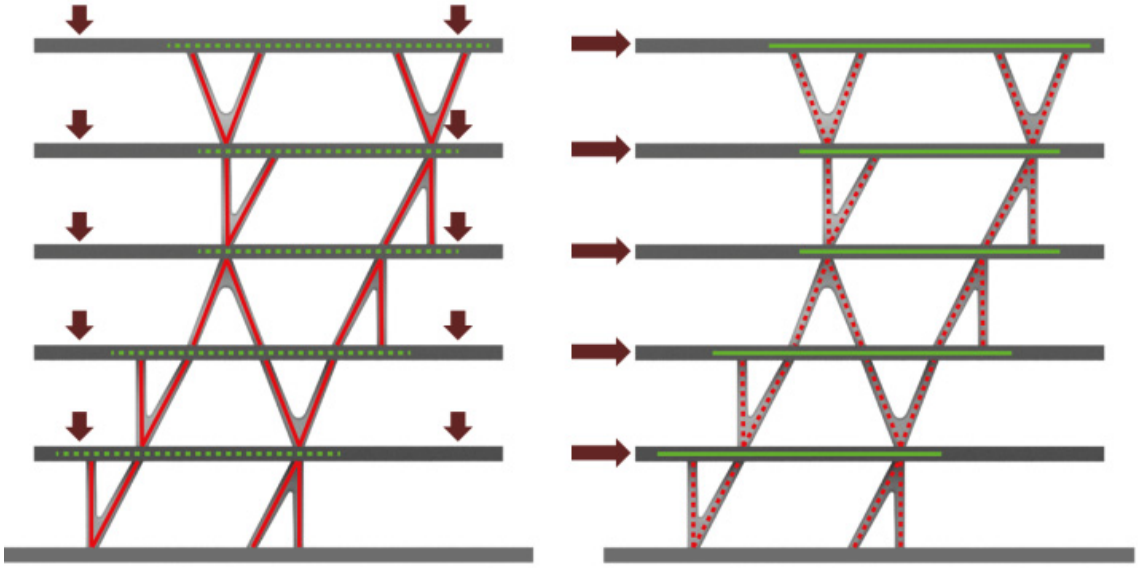
Referenční stavba

Téma diplomové práce je inspirováno stavbou v lichtenštejnském městě Vaduz od architektů Falkeis Architects. Stavba nese název **Active Energy Building** a jedná se o bytový dům s důrazem kladeným na udržitelnost. Aspekt udržitelnosti se projevuje v rámci celé budovy a ovlivňuje tak její vzhled, funkčnost i nosný systém. Hlavním cílem architektů byla energetická soběstačnost a nezávislost na fosilních palivech. Budova proto využívá pouze obnovitelné zdroje energie a funguje jako lokální solární elektrárna, která přebytek elektrické energie dokáže distribuovat do dalších čtyř napojených budov v okolí. Dominantou budovy je trojrozměrná Voroného teselace, která slouží jako nosný prvek pro aktivní energetické systémy budovy. Jednotlivé buňky Voroného diagramu vyplňují solární panely a speciální panely nazývány „climate wings“, které jsou součástí vzduchotechniky a umožňují využít nashromážděné teplo k ohřívání vzduchu nebo naopak chladný vzduch k ochlazování. Solární panely mají navíc možností vyklápění a natáčení, a proto dokážou sledovat polohu slunce a maximalizovat solární zisky.

Obrázek 4.1: Active Energy Building[16]

4.1 Konstrukční systém stavby

Do udržitelnosti budovy přispívají nejenom její aktivní energetické systémy a z toho pramenící ekonomičnost a efektivita provozu, ale i schopnost adaptace a možnost změny funkce budovy. To byly jedny z hlavních podmínek při navrhování hlavního nosného systému budovy. U tradičních bytových domů se nosná konstrukce odvíjí od parkovacích stání v garážích a tím přímo ovlivňuje tvar, velikost a dispozice bytových jednotek. Od předdefinovaného rozložení nosných prvků se architekti chtěli odpoutat, a tak vznikl systém sloupových prvků ve tvaru písmene V nebo A, který dokáže vhodným uspořádáním sloupů zredukovat vliv rastru parkoviště a vytvořit libovolné dispozice ve vyšších podlažích.[17]



Obrázek 4.2: Přenášení svislých a vodorovných sil [18]

K dosažení co největší volnosti v prostoru je také vhodné minimalizovat počet stěnových prvků. V budově je jen jedna nosná stěna vnější a vnitřní nosné stěny tvoří pouze jádro výtahové šachty. Dělící konstrukce jsou pak v celém objektu zhotoveny z lehkých materiálů, které je možné v průběhu životnosti stavby pozměnit bez vlivu na statiku budovy.

S minimálním použitím stěn nastává problém menší tuhosti celé stavby, a to hlavně ve vodorovném směru. I proto byly zvoleny sloupy tvaru A nebo V, které dokáží spolupůsobením s deskami budovu dodatečně ztužit a vytvořit svým tvarem efektivní náhradu smykových stěn. Pro přenášení sil ve svislém směru je využíváno napojování sloupových prvků a vytváření nosných vazeb skrze několik podlaží, které přispívají ke globální tuhosti stavby.

4.1.1 Sloupové prvky

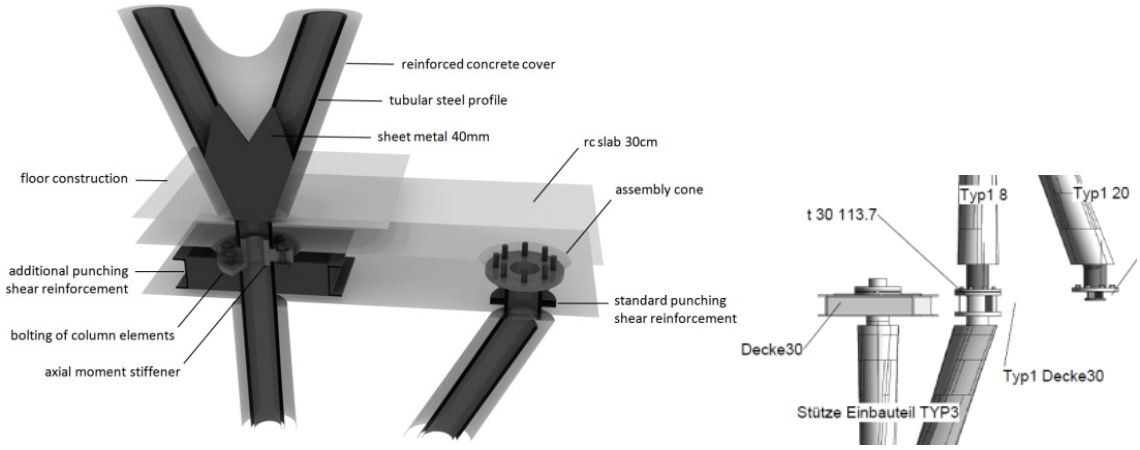
Obrázek 4.3: Varianty sloupů [18]

Za návrhem sloupových prvků stojí kromě architektů i zmíněná statická kancelář Bollinger+Grohmann, která se často věnuje neobvyklým, architektonicky zajímavým stavbám a využívá inovativní a moderní přístupy k analýze budov. Jedním z úkolů B+G bylo do budovy umístit co nejmenší počet sloupů a vyvážit malý počet vhodným umístěním, geometrií a větvením sloupů. K tomu byly navrženy čtyři sloupy, dva sloupy tvaru V a dva sloupy ve tvaru A, z toho jeden symetrický a druhý asymetrický viz Obrázek 4.3. Dvě varianty sloupů od každého typu mají zaručit maximální možnost kombinací při zachování malého počtu prvků [18]. Klíčové pro ekonomičnost a zároveň design je právě využití prefabrikace, kde vznikali použité ocelobetonové sloupy.

Obrázek 4.4: Prefabrikované sloupy [18]

4.1.2 Materiál a spoje

Jádro sloupu tvoří ocelové trubky, které se sbíhají do styčníkové ocelové desky tl. 40 mm a pro napojení jsou určené koncovky ocelové trubky vystupující ze sloupu. Vnější část tvoří beton, který zde slouží jako protipožární ochrana a plní funkci estetickou. Pro napojení sloupů slouží speciální spojovací prvky typu kruhové desky s dírami pro šrouby. Na hlavy sloupů se dle rizika protlačení montuje buď standardní výztuha nebo výztuha vylepšená proti protlačení viz Obrázek 4.5a. Na stejném obrázku vpravo je též znázorněna vazba se stropní deskou, kde styčná deska musí zůstat po betonáži stropu odhalená a dostupná pro montáž navazujícího sloupu.



(a) (b)

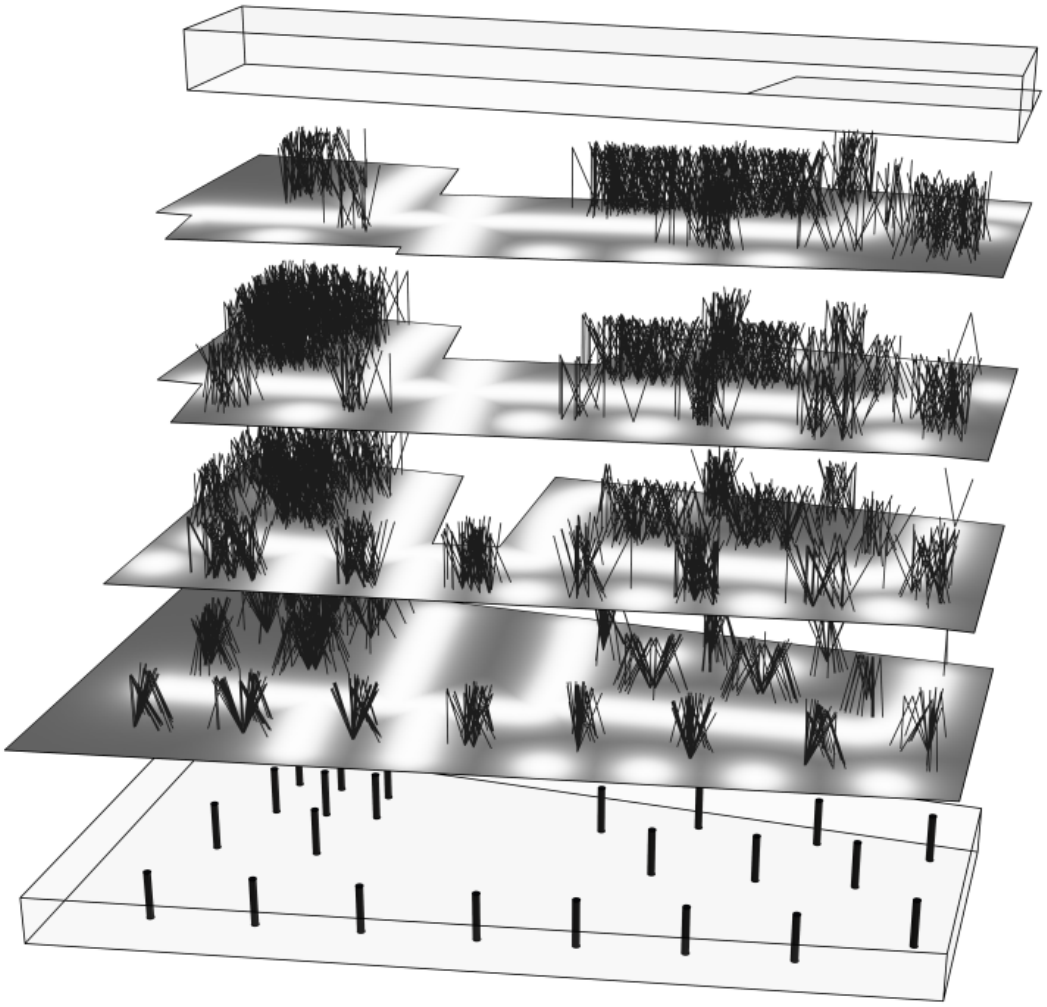
Obrázek 4.5: Napojení sloupů, upraveno podle [18]

4.2 Optimalizace konstrukčního systému

Pro návrh a optimalizaci sloupových prvků autoři využívali hlavně parametrického modelování v programu Rhino a Grasshopper spolu s rozšiřujícími balíčky Karamba3D a Galapagos. Taktéž využili algoritmu BESO upraveného přesně na tento typ optimalizace. Inženýři z B+G konstrukci optimalizovali ve dvou krocích. V prvním kroku byla optimalizována topologie konstrukce, tedy rozložení sloupů v návrhovém prostoru a ve druhém kroku byla optimalizována geometrie formou hledání ideální orientace/rotace každého sloupového prvku.

4.2.1 Optimalizace topologie

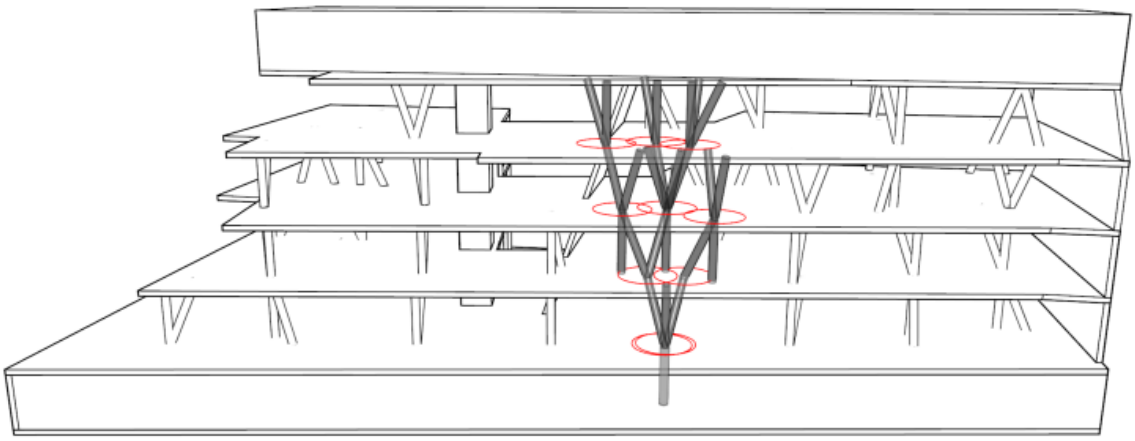
Na začátku procesu byly zvoleny prostory, v kterých dle architektonický požadavků nebylo vhodné umístit sloupy. Tímto omezením tak vznikl v každém podlaží tzv. návrhový prostor, který určoval potenciální polohu sloupů. V dalším kroku inženýři aplikovali algoritmus, který měl za úkol zaplnit návrhový prostor variantami sloupů. Celkově bylo v budově vygenerováno 3650 sloupových prvků viz Obrázek 4.6. Následně bylo využito upraveného BESO algoritmu, pomocí kterého byl počet zredukován na 27 sloupových prvků. Před začátkem druhého kroku byly vybrané sloupy ještě ručně upraveny, protože nebyly na sobě přesně navázány. [18]



Obrázek 4.6: Návrhový prostor pro topologickou optimalizaci [18]

4.2.2 Optimalizace geometrie

Druhým krokem byla optimalizace geometrie, přesněji orientace sloupů, pomocí genetického algoritmu Galapagos, což je jednoduchý GA, který je součástí rozhraní Grasshopper a umožňuje pouze jedno-kriteriální optimalizaci. Z tohoto důvodu byla kritéria optimalizace (deformace, poloha sloupů v návrhovém prostoru a v budově, využití sloupů) sloučena do jednoho. Jako geny algoritmu byly zvoleny parametry rotace každého sloupu, resp. pro každý sloup byl pouze jeden parametr rotace kolem počátečního (napojeného) bodu sloupového prvku. Následně genetický algoritmus dokonvergoval k optimálnímu řešení s nejmenší deformací.



Obrázek 4.7: Znázornění parametru rotace každého sloupu [18]

Jelikož byla používána jedno-kriteriální optimalizace, tak inženýři neměli možnost vybírat z množiny Pareto optimálních řešení, a tak nalézt více variant a následně analyzovat více výsledků. Sami autoři popisují, že při návrhu budovy v roce 2011 ještě nebyly dostupné takové možnosti, jako např. nástroje vícekriteriální optimalizace v prostředí Grasshopper a další.

Prostudování práce R. Vierlingera, A. Hofmann a K. Bollingera [18] o návrhu Active Energy Building přineslo do diplomové práce velký vhled na celou problematiku optimalizace sloupů a konstrukčního systému a práce byla také inspirací při vývoji vlastního algoritmu pro generování a optimalizaci sloupových prvků.

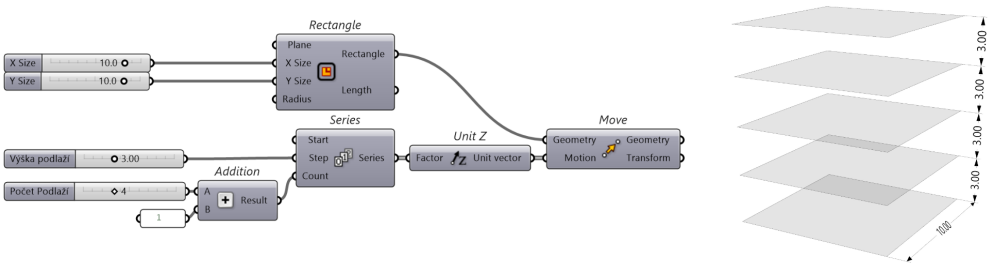
Kapitola 5

Vývoj algoritmu generování sloupů

Kapitola se zabývá vývojem a testováním navrženého algoritmu na optimalizaci sloupů. Pro vytváření a testování algoritmu na hledání vhodné polohy a tvaru sloupů, bylo užitečné vytvořit jednoduchý a menší model, na kterém lze všechny funkce modelu rychle, efektivně zkoušet a výsledky věrohodně ověřovat. Tento model bude dále v práci zmiňován jako zkušební model. Optimalizační algoritmus využívá program na analýzu konstrukcí Karamba3D a evoluční řešič WallaceiX. Oba programy jsou rozšiřující balíčky pro prostředí Grasshopper a jsou popsány v kapitole o parametrickém navrhování v sekci 2.1. Příklady využití a práce s rozšířením je popsána v následujících sekcích.

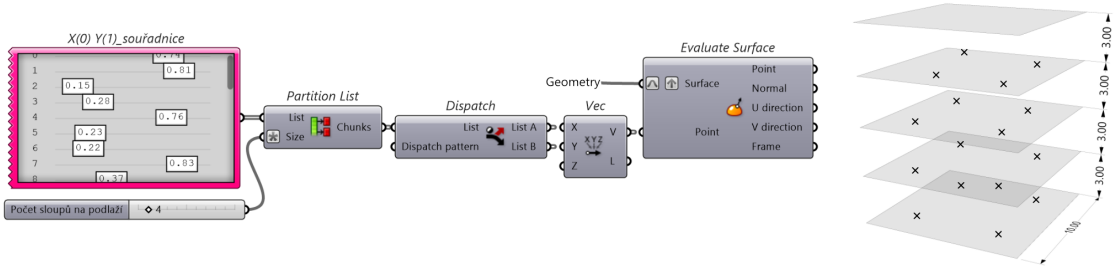
5.1 Tvorba geometrie modelu

Úloha byla sestavena a následně testována na zkušebním modelu, který sestává z čtvercových ploch reprezentujících stropní desky. Vytvořit takový zkušební model je v prostředí Grasshopper velmi jednoduché. Postup k vytvoření několika desek vzdálených od sebe určitou vzdálenost je znázorněn na Obrázku 5.1.



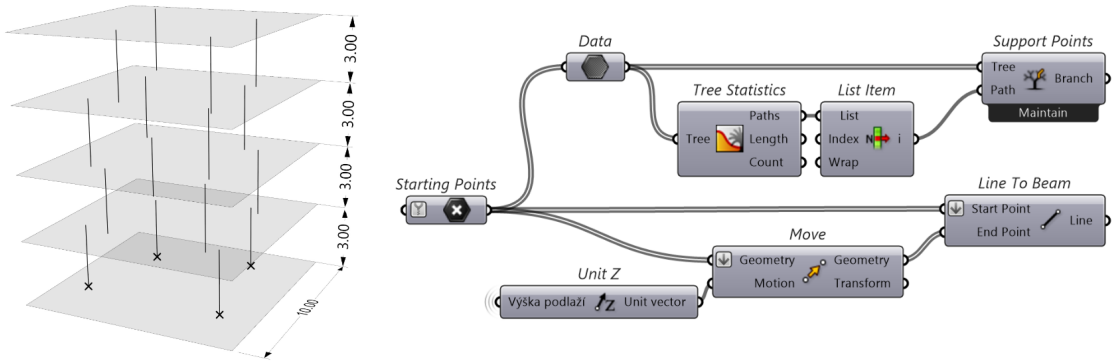
Obrázek 5.1: Ukázka tvorby základní geometrie v GH

Dále je potřebné vytvořit body, z kterých se budou tyčit jednotlivé sloupy. Tyto body budou nazývány startovací. K vytvoření startovacích bodů použijeme plochy desek, které už máme vytvořené. Jednoduchý postup je opět znázorněn Obrázkem 5.2, kde do komponenty *Evaluate Surface* vstupují vytvořené plochy. Do vstupu pro bod je pak zapojen vektor s X a Y souřadnicí, která je navolena v *Gene Pool* slideru. Souřadnice XY nabývají hodnot od 0 do 1, protože jednotlivé plochy jsou reparametrizovány do tohoto intervalu komponentou *Evaluate Surface*.



Obrázek 5.2: Ukázka vytváření bodů na desce

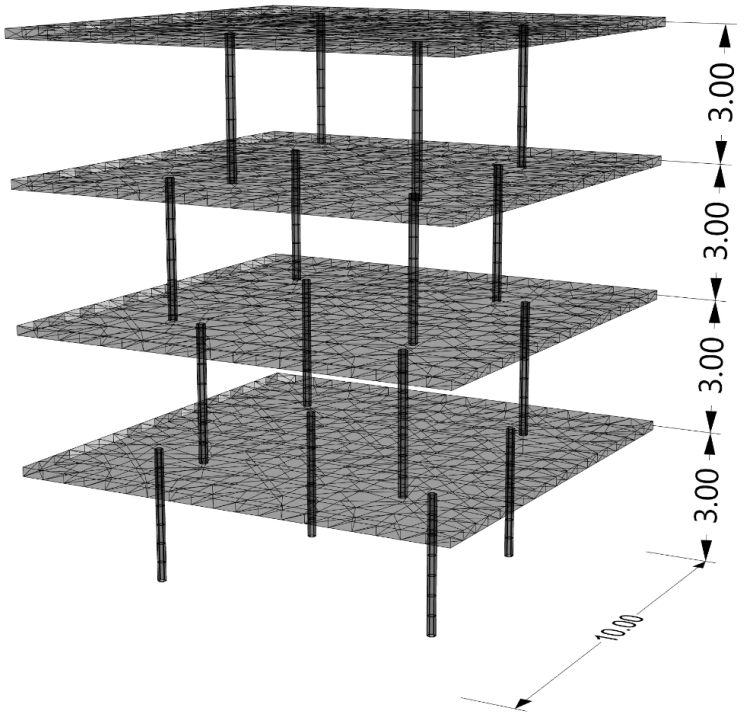
Nyní stačí startovací body posunout vždy o podlaží výše, čehož docílíme použitím komponenty *Move* s vektorem ve směru Z o hodnotě výšce podlaží. V dalším kroku spojíme právě vytvořené body s body startovacími. Je důležité u obou množin bodů zachovat jejich řazení tzv. „Paths“ , což jsou jednotlivé datové cesty. Data bodů pak budou v našem případě rozřazena do čtyř větví dle čtyř desek o čtyřech bodech. V tomto okamžiku máme vytvořené linie, které už můžeme použít jako vstup pro vytvoření sloupů např. v Karamba3D. Pro dokončení přípravy vstupů ke statické analýze potřebujeme ještě podpory. Ty vytvoříme z bodů na nulté desce tak, že vybereme odpovídající větev bodů pomocí *Tree Branch*.



Obrázek 5.3: Ukázka tvorby křivek pro sloupy

5.2 Analýza modelu

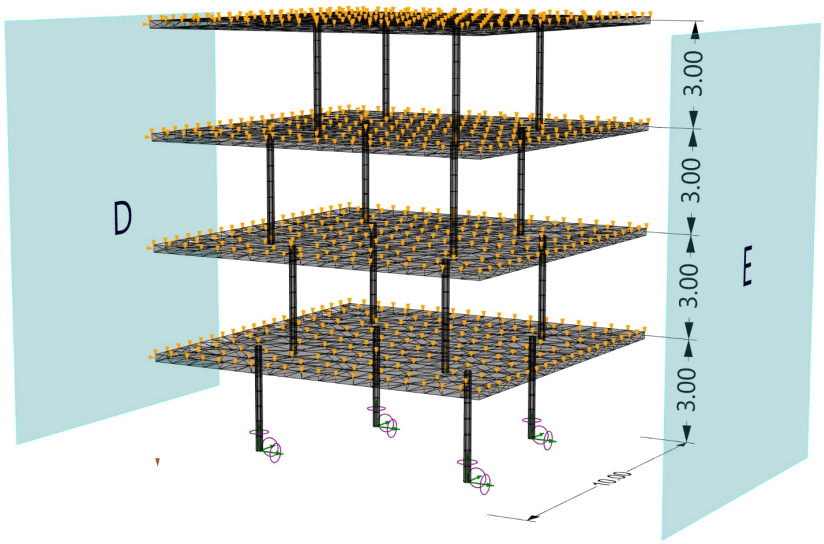
Statická analýza bude provedena v rozšíření Karamba3D, kde nejdříve musíme sestavit model pro MKP analýzu programem. V geometrickém modelu máme předpřipravené plošné prvky pro desky, liniové prvky pro sloupy a body pro podpory. Plošné prvky je nutné pro MKP analýzu přetransformovat do sítě, což lze jednoduše komponentou *Mesh Brep*, kde také nastavíme velikost sítě, zjemnění na okrajích sítě nebo vložíme vrcholové body sloupů do sítě. Sloupy vytvoříme komponentou *Line To Beam*. Velikosti sítí lze libovolně měnit a změnu ve výsledcích vidět v reálném čase. To přináší rychlou zpětnou vazbu při testování funkčnosti a věrohodnosti výpočtu. Síť konečných prvků byla pro začátek nastavena na tloušťku desky. Pro desky i sloupy doplníme ještě průřezové vlastnosti. Sloup byl zvolen plného kruhového průřezu s průměrem 30 cm, tloušťka desky byla zvolena 20 cm a materiál pro oba prvky byl určen beton třídy C30/37.



Obrázek 5.4: Zasíťovaný model

5.2.1 Zatížení

Pro účely vývoje a návrhu v rané fázi projektu budou zatěžovací stavy zjednodušeny a upraveny podle potřeb algoritmu.



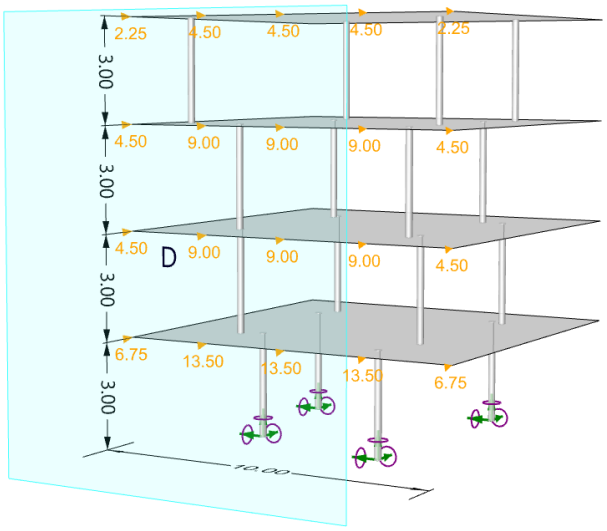
Obrázek 5.5: Zatížený model

Svislé zatížení

Do svislého zatížení je zahrnuta vlastní tíha konstrukce spolu s kombinací zatížení od podlahy a užitného zatížení. Pro zatížení vlastní tíhou v Karamba3D lze použít komponentu *Loads* s typem zatížení „gravity“, kde je vektor působení zatížení nastaven na {0,0,-1}, tedy ve většině případů správné nastavení s gravitací svisle dolů. Hodnotu zatížení komponenta převezme ze zadaného materiálu, kde u betonů je objemová hmotnost stanovena na $2500 kg/m^3$. Kromě vlastní tíhy je na konstrukci zadáno zmiňované plošné zatížení, které reprezentuje zatížení od podlahy spolu s užitným zatížením. Hodnota zatížení je nastavena na $5 kN/m^2$.

Vodorovné zatížení

Vodorovné zatížení desek představují síly od větru, které jsou na konstrukci zadány pomocí svislých plošných zatížení se silami ve směru osy X nebo Y. Maximální dynamický tlak byl vypočten pomocí tabulkového procesoru Excel dle normy ČSN EN 1991-1-4. Vypočtená hodnota dynamického tlaku větru pro zkušební model je $0.965 kN/m^2$. Pro návětrnou stranu, oblast D, je stanoven součinitel $c_{pe} = 0.80$ a tedy výsledná hodnota zatížení je $0.77 kN/m^2$ ($\doteq 0.8 kN/m^2$). Po vynásobení bezpečnostním součinitelem $\gamma_q = 1.5$ je $q_p = 1.2 kN/m^2$.



Obrázek 5.6: Rozložení zatížení větrem na stropní desky

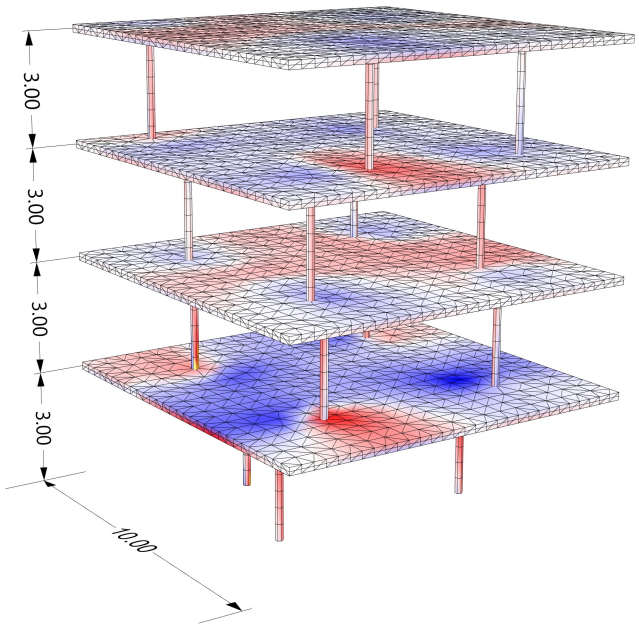
Správnou distribuci zatížení z plochy na jednotlivé body desky lze ověřit dle Obrázku 5.6. Lze předpokládat, že na zkušební model bude osazen celistvý vnější plášť a síly působící na obvodový plášť budou přenášeny stropními deskami. Na ukázku má každá deska na okraji pět uzlů sítě konečných prvků vzdálených od sebe $2.5m$. Každá z prostředních dvou desek má zatěžovací pole o výšce podlaží, tedy $3m$ a na celou stranu působí tlak výše vypočtený $\mathrm{q_p} = 1.2kN/m^2$. Pak lze hodnotu síly v uzlu vypočítat jako:

$$f = q_\mathrm{p}[kN/m^2] \cdot 3[m] \cdot 2.5[m] = 9[kN]$$

Výsledek se shoduje s hodnotami na Obrázku 5.6 v mezilehlých deskách. Pro spodní a vrchní desku platí jiné zatěžovací šířky a to $4.5m$, resp. $1.5m$.

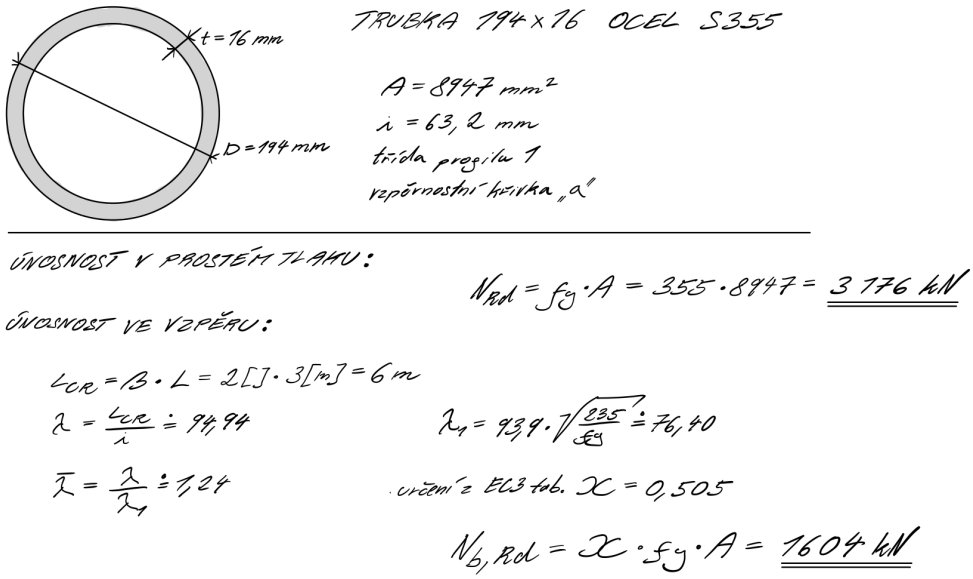
5.2.2 Výsledky analýzy

Výsledky analýzy pomocí aplikace Karamba3D je možné vidět ihned po zapojení vhodného řešiče a komponenty na zobrazení výsledků. Ve většině případu se využívá řešič prvního řádu a k zobrazování výsledků komponenty *Model View* , *Beam View* nebo *Shell View*. Pomocí komponenty na zobrazení výsledků na prutových prvcích lze zobrazit: vnitřní síly, osové napětí, deformace nebo využití prvku. Mezi základní výsledky na plošných prvcích patří: deformace, hlavní napětí 1 a 2 nebo využití prvku. Pro představu rozložení napětí a dimenzovatelnosti konstrukce v raném návrhu je často využíváno zobrazení využití (utilization), což je poměr mezi napětím a pevností materiálu.



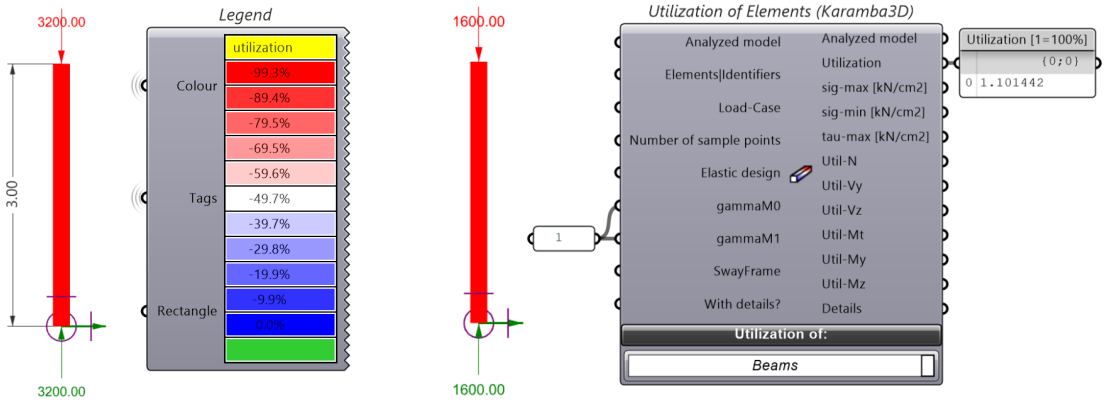
Obrázek 5.7: Napětí na konstrukci zobrazené pomocí využití prvků

Ověření výsledků lze provést například na sloupu, který bude uvažován jako v referenční stavbě ocelobetonový, kde sloup tvoří obetonovaná ocelová trubka. Ve výpočtu a analýze je tedy uvažováno s ocelovou trubkou průměru 194 mm a tloušťkou stěny 16 mm. Pro výpočet vzpěru je sloup uvažován jako konzola.



Obrázek 5.8: Výpočet únosnosti trubky

Vypočtená únosnost v prostém tlaku je zhruba $3200kN$ a zatíženy sloup takovou silou vykazuje využití 100%. Odchylky jsou způsobené nepřesností ve výpočtu. Vypočtená únosnost v tlaku s vlivem vzpěru vychází na $1600kN$. Využití se zahrnutím vzpěru se získá pomocí komponenty *Utilization of Elements* a jeho hodnota činí 110%. Oproti ručnímu výpočtu je návrh z Karamba3D konzervativnější.



Obrázek 5.9: Využití trubky - prostý tlak a vzpěr

Princip výpočtu využití u plošných prvků je obdobný, akorát posuzované napětí je určováno dle pevnostních hypotéz. Pevnostní hypotézu si uživatel může změnit v nastavení materiálu. Pro izotropní materiály jsou doporučeny následující hypotézy: Von Mises, Tresca, Rankine. Pro ortotropní materiály je navíc k výběru Tsai Wu hypotéza. Výchozí nastavení je například pro ocel Von Mises a pro beton Rankine.

5.3 Parametry optimalizace

5.3.1 Geny

K optimalizaci pomocí genetického algoritmu je potřebné zadat základní vstupní údaje. První údaj jsou proměnné hodnoty, nazývané geny, které charakterizují a vytváří jedince v populaci. Geny v případě zkušebního modelu tvoří:

- XY souřadnice sloupu
- Rotace sloupu
- Typ sloupu

Rotace a typ sloupu jsou zavedeny jako geny pro sloupové prvky ve tvaru A nebo V. Pro zkušební příklady s klasickými sloupy tyto geny nejsou uvažovány.

5.3.2 Kritéria

Druhý zásadní vstupní údaj jsou kritéria, dle kterých bude evoluční algoritmus ohodnocovat jednotlivé jedince. Evoluční řešič jednotlivá kritéria vždy minimalizuje. Tyto kritéria jsou pro výsledek a správné fungování optimalizace velmi podstatné, a proto je nutné důkladné otestování a odladění jednotlivých kritérií. Kritéria používané ve zkušebním modelu jsou následující:

K1 - Deformace

Hlavní kritérium pro testování modelu. Maximální deformace jsou vypočtené ke každému typu zatížení a dle potřeb testování se toto kritérium buď rozděluje na svislé a vodorovné deformace, nebo se deformace sčítají do jedné hodnoty.

K2 - Využití sloupů

Využití všech sloupů se buď sčítá do jednoho kritéria, nebo je vypočten rozdíl využití sloupů v každém podlaží a jednotlivé hodnoty se sčítají do jedné. Druhý způsob s rozdílem využití má zajistit rovnoměrné rozložení sloupů.

K3 - Využití desek

Součet využití všech desek ve všech bodech sítě, nebo konkretizace kritéria na maximální hodnotu využití v každé desce a popř. jejich průměr.

K4 - Kolize

Kritérium udává počet koncových bodů sloupu, které leží mimo desku nebo v chráněné oblasti uvnitř desky.

K5 - Rozložení

Pomocné kritérium pro získání většího rozestupu mezi sloupy. Algoritmus vyhodnocuje vzdálenosti mezi sloupy v rámci každého podlaží a nejmenší vzdálenost je předána evolučnímu řešiči jako kritérium, které má maximalizovat. Jelikož řešič kritéria vždy minimalizuje, je nutná úprava hodnoty $K5$ na hodnotu $1/K5$.

5.3.3 Parametry algoritmu

Pro zkušební model se osvědčilo nastavení řešiče znázorněné v tabulce níže. Dle složitosti příkladu se samozřejmě měnily hodnoty populace, zejména se pak navyšoval počet generací. Parametry algoritmu jako pravděpodobnost mutace mezi 0.01 a 0.1 nebo nižší distribuční indexy přinášely většinou lepší řešení.

Populace	Velikost generace	50
	Počet generací	50
Parametry algoritmu	Pravděpodobnost křížení	0.9
	Pravděpodobnost mutace	0.1
	Index distribuce křížení	5
	Index distribuce mutace	5
	Náhodný seed	0

Tabulka 5.1: Nastavení algoritmu

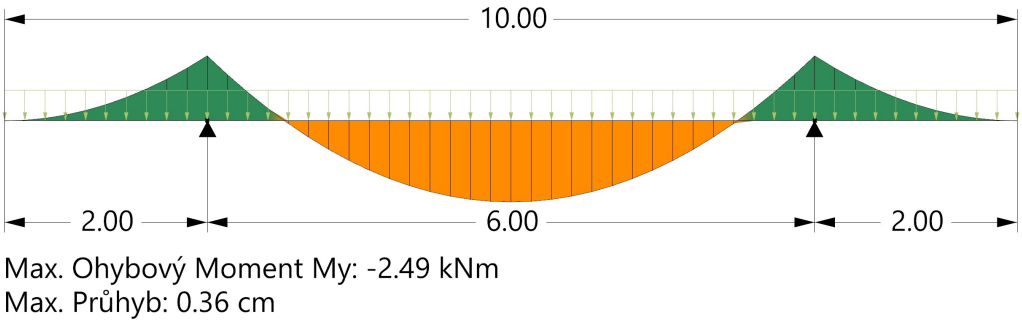
5.4 Výstupy z testování na jednopodlažním modelu

Výstupy z jednotlivých zkoušek jsou uvedeny dle pořadí od nejjednodušších příkladů až po ty složitější. Stejným způsobem bylo samotné testování prováděno a k dalším fázím zkoušek se přistoupilo až po dosažení věrohodných výsledků na příkladech jednodušších. Pro účely testování byly optimalizace prováděny v malém měřítku s větším důrazem na rychlost a kvalitu konvergence nežli na přesné výsledky optimalizace.

5.4.1 Rovné sloupy

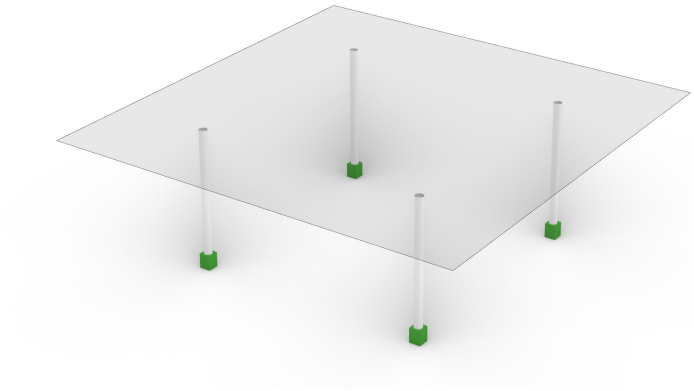
Svislé zatížení

První zkouškou bylo testování rozmístění klasických rovných sloupů se svislým zatížením. Sloupy se mohou pohybovat kdekoliv po desce a jejich výsledné rozmístění závisí čistě na volbě kritérií a nastavení algoritmu. Odhad výsledného rozmístění sloupů byl podpořen zjednodušením příkladu na spojitě zatížený nosník a následném nalezení nejoptimálnější polohy podpor. Nejmenší průhyb nastává, pokud jsou podpory umístěny přibližně tak, aby ohybový moment byl na nosníku rozložen rovnoměrně. Nejoptimálnější umístění podpor je v tomto případě přesně 2/9 délky nosníku od krajů (níže jsou podpory umístěny do 1/5 délky).

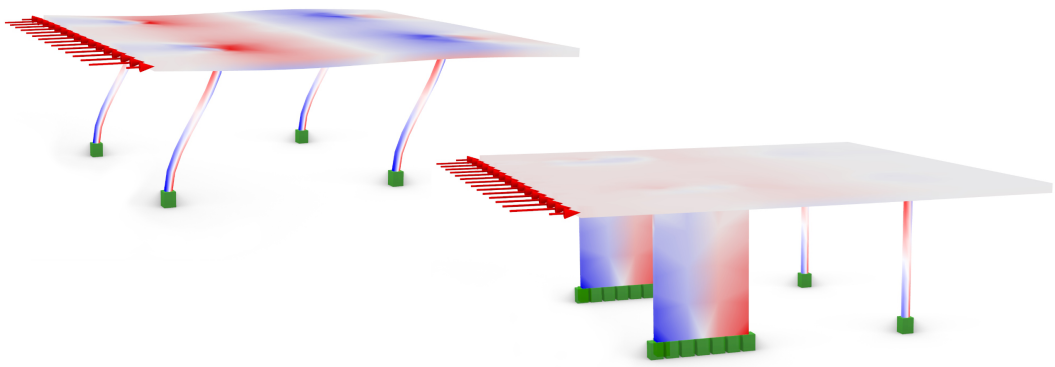


Obrázek 5.10: Zkouška spojitě zatíženého nosníku

Optimalizace rozmístění sloupů proběhla dle následujících kritérií: K1 - deformace, K2 - využití sloupu, K3 - využití desky. Výsledná poloha sloupů viz Obrázek 5.11 je velmi blízká zjištěné optimální poloze viz Obrázek 5.10. Odchylka od polohy, kde jsou hodnoty deformace nejmenší je v řádech centimetrů a poloha sloupů k dané hodnotě rychle konverguje.



Obrázek 5.11: Příklad výsledku optimalizace - typ I



Obrázek 5.12: Deformovaná konstrukce v důsledku vodorovného zatížení

Vodorovné zatížení

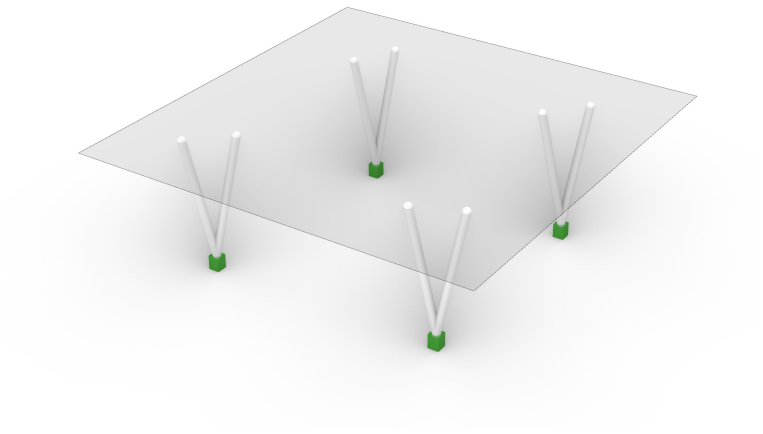
Při vodorovném zatížení konstrukce s klasickými sloupy nedosahuje velké tuhosti a podléhá velkým deformacím. Větší vodorovné tuhosti by se dalo dosáhnout pomocí ztužujících smykových stěn, např. umístěných v rámci funkčního jádra.

5.4.2 Sloupy ve tvaru V nebo A

Následuje testování šikmých sloupů ve tvaru V nebo A, při kterých se již nehledá pouze poloha sloupů, ale také vhodná rotace a typ prvku.

Svislé zatížení

U svislého zatížení se dá očekávat, že algoritmus zvolí sloupy typu V, které nabízí stropní desce více podpůrných bodů a v důsledku toho bude menší průhyb i využití desky.

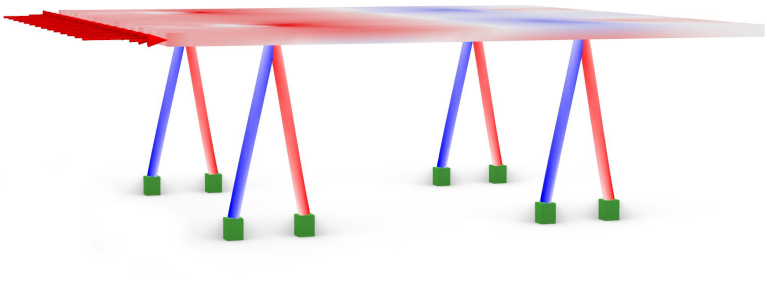


Obrázek 5.13: Výsledek optimalizace sloupů tvaru A / V, svislé zatížení

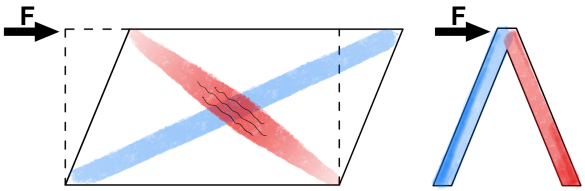
Optimalizace proběhla opět dle kritérií K1, K2, K3. Konvergence k správné poloze a typu prvku byla rychlá. Rotace prvků byla očekávána směrem do diagonálních směrů, nicméně rozdíly ve využití nebo v deformacích jsou malé.

Vodorovné zatížení

U horizontálně zatížené konstrukce algoritmus správně vybere typ prvku A, který dosahuje větší tuhosti ve vodorovném směru a funguje obdobně jako diagonální ztužení nebo smyková stěna, viz Obrázek 5.15.



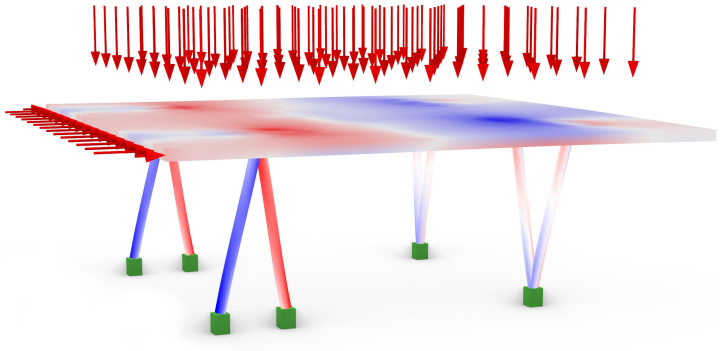
Obrázek 5.14: Výsledek optimalizace sloupů tvaru A / V, vodorovné zatížení



Obrázek 5.15: Schéma ztužení a přenosu sil

Kombinace vodorovného a svislého zatížení

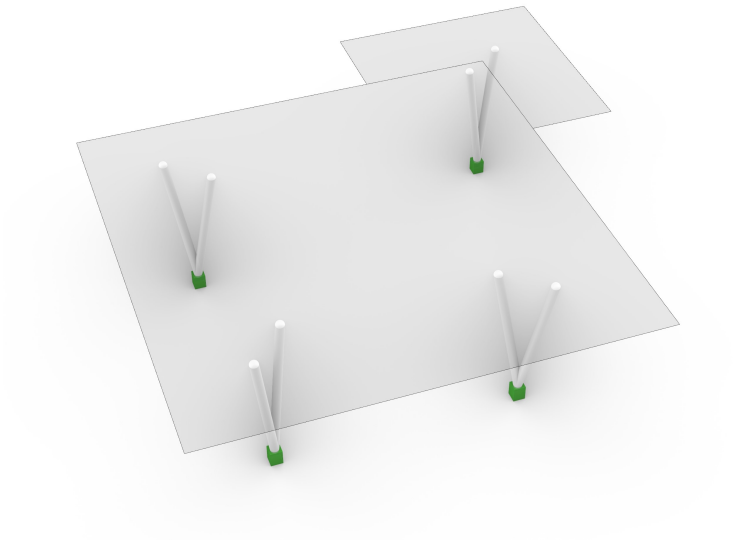
Na konstrukci v tomto testu působí zároveň vodorovné a svislé zatížení. Lze předpokládat určitou kombinaci sloupů typu A, které budou působit proti účinkům vodorovného zatížení a sloupů typů V, které budou zajišťovat menší svislé deformace.



Obrázek 5.16: Výsledek kombinace H+V zatížení

Vykonzolovaná část desky

Další zkouškou je přidání rohového balkonu na desku a reakce sloupů na změnu tvaru desky. Zahrnuto je pouze svislé zatížení z důvodu snadnější ověřitelnosti výsledků.



Obrázek 5.17: Výsledek optimalizace sloupů na desce s balkonem

Odezva sloupových prvků na změnu geometrie a zatížení je opět znatelná. Dle výsledků optimalizace lze vidět posunutí podpory do míst balkonu k vyvážení svislé deformace na desce. Na výsledcích je také vidět natočení sloupů v poli do diagonálních směrů desky z důvodu snížení deformací v prostředním poli desky.

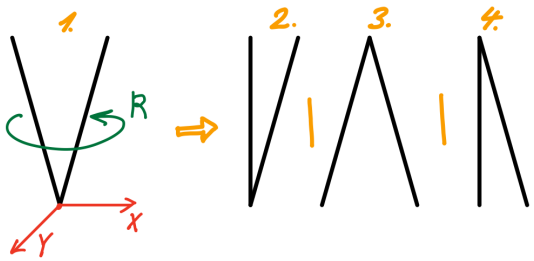
5.5 Řešení vícepodlažního modelu

Po úspěšném testování algoritmu na jednopodlažním modelu je přistoupeno k modelu vícepodlažnímu. S narůstajícím počtem podlaží narůstá také počet jednotlivých genů a parametrů algoritmu, což negativně ovlivňuje výpočetní dobu každé optimalizace. Nejenom z tohoto důvodu budou v této sekci popsány další nové přístupy a vylepšení algoritmu, které mají za úkol zrychlit výpočet a zefektivnit generování sloupových prvků.

5.5.1 Volné generování sloupů

Tento přístup generování sloupů vychází z předchozího testování na jednopodlažním modelu, kde se jednotlivé sloupy mohou generovat kdekoliv na zvolené ploše a jejich vlastnosti jsou řízené pouze kritérii evolučního řešiče. Výhodou volného generování na ploše je prohledání celého návrhového prostoru, a tudíž je větší pravděpodobnost k nalezení optimální polohy sloupu. Nevýhodou je pak velké množství genů, které vstupují do evolučního řešiče. Každý sloup je definován XY souřadnicemi základny, rotací a typem sloupového prvku stejně jako v předchozí sekci. Další nevýhodou je

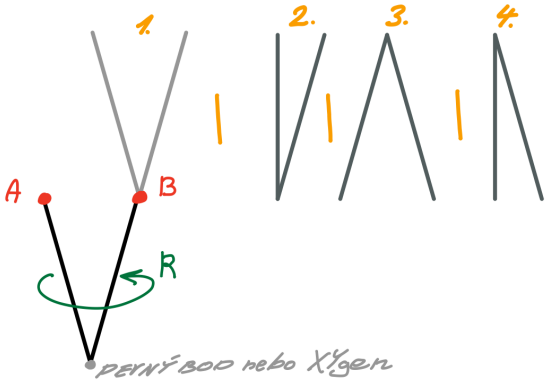
napojování prvků, které není řešeno v algoritmu generování sloupů a je využíváno pouze kritéria, které vstupuje do evolučního řešiče, a má na starost minimalizovat vzdálenost koncového bodu sloupu a počátečního bodu sloupu v rámci každé desky. Toto kritérium napojování sloupů se během testování ukázalo jako nespolehlivé, jak z důvodu nepřesnosti napojení, tak i pomalé konvergence kritéria.



Obrázek 5.18: Znázornění genomu volného generování sloupů

5.5.2 Strukturální generování sloupů

Metoda strukturálního generování sloupů vznikla kvůli chybějící funkci napojování sloupů v předchozí metodě. Jak už samotný název vypovídá, jde o přístup, kde následující sloupy mohou vznikat pouze v koncových bodech sloupech předchozích. Metoda již sama o sobě vytváří sloupové větvě a algoritmus sloužící ke generování sloupů má za úkol vybírat vhodné body k napojení dalšího prvku.



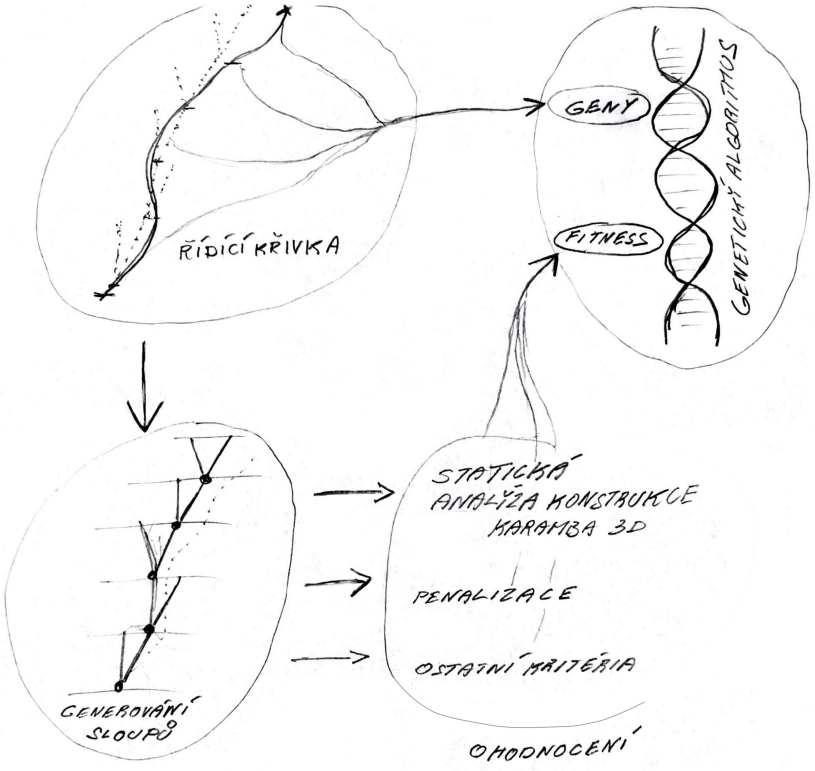
Obrázek 5.19: Znázornění genomu strukturálního generování

Komponenty algoritmu pro každé podlaží musí být zapojeny sériově, tedy od nejnižšího podlaží až po nejvyšší. K tomu lze využít pluginy jako je např. Anemone, Hoopsnake nebo OctopusLoop, pomocí nichž lze vytvořit smyčku, která dokáže přenášet data o sloupech mezi podlažími. Problém s pluginy tohoto typu je, že nejsou optimalizovány na použití s evolučními řešiči. Jediný, který lze použít je

OctopusLoop, který je součástí evolučního řešiče Octopus. V diplomové práci je využíváno řešiče Wallacei, který bohužel ani s OctopusLoop smyčkou nefungoval správně, a proto je nutné použít sériového opakování komponent, což lze vyhodnotit jako nevýhodu přístupu. Další nevýhodou tohoto přístupu je stále velké množství genů a s tím spojený velký počet řešení.

5.5.3 Generování pomocí vodící křivky

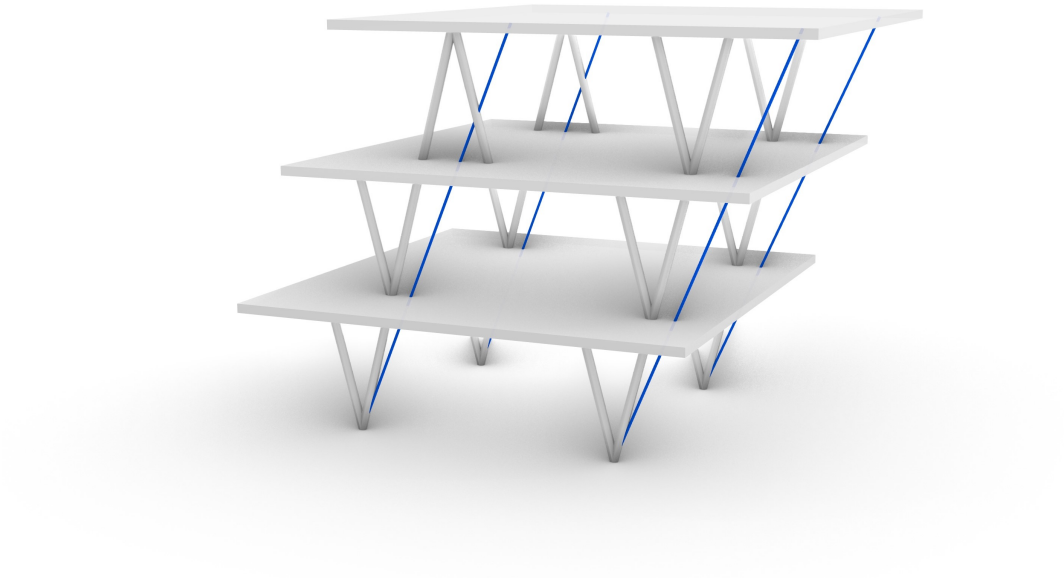
Metoda vodící, nebo také řídící křivky je vylepšení předchozí metody strukturálního generování sloupů, u které zůstávalo nevýhodou velké množství genů vstupující do evolučního řešiče. Zjednodušený princip této metody spočívá ve vytvoření vodící křivky ke každé sloupové větvi (větev - řada vertikálně navazujících sloupů) a následné generování sloupů dle vlastností křivky v určitých bodech. Do evolučního řešiče vstupují jako geny pouze ty parametry, které definují křivku samotnou, nikoliv sloupové prvky. Jednotlivá řešení křivky jsou již ohodnocena dle kritérií vycházejících z analýzy celé konstrukce včetně samotných sloupových prvků. Tímto způsobem se upravuje pouze genom křivky, z kterého se generují varianty sloupů podle ohodnocení konstrukce včetně penalizačních faktorů.



Obrázek 5.20: Schéma principu řídící křivky

Lineární křivka

Vodící křivku lze vytvořit mnoha způsoby a může nabývat nejrůznějších tvarů. Lineární křivka má výhodu jednoduché tvorby a snadné kontroly tvaru křivky pomocí malého počtu parametrů. Na Obrázku 5.21 je vidět vodící křivka v podobě přímky procházející skrze všechny podlaží. Tato křivka má pevně ukotvený základní bod a lze měnit pouze její vodorovné souřadnice v rámci nejvyšší desky. Jednotlivé sloupy se generují a rotují dle orientace a polohy vodící křivky v každém podlaží. Výhodou je tak rychlá odezva změny sloupových prvků na jednoduchý tvar křivky a malý počet genů přímky vstupující do evolučního řešiče. Jednoduchost křivky je nicméně i její nevýhoda, protože přímka sama o sobě nenabízí dostatečný počet parametrů pro algoritmus generování sloupových prvků. Dodatečné parametry (geny) například pro volbu typu sloupu nebo nezávislou rotaci musí být vytvořeny uměle, což nesplňuje očekávání této metody. Z toho důvodu bylo přistoupeno k složitějším křivkám.

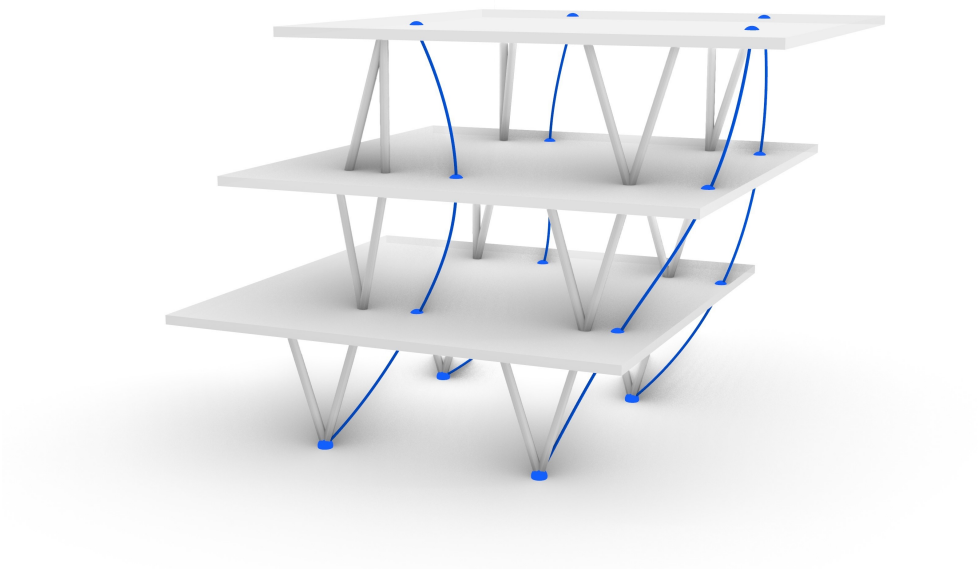


Obrázek 5.21: Generování sloupů dle lineární křivky

Interpolace křivkou vyššího stupně

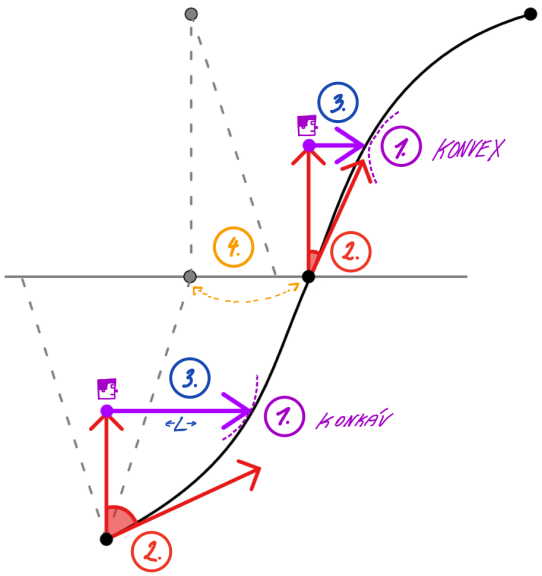
K vytvoření složitější křivky lze přistoupit více způsoby. Nejjednodušší je využít nativní komponenty prostředí Grasshopper, které pro složitější křivky nabízí funkce jako: *Interpolate*, *NURBS Curve*, *Tangent Curve* aj. Například pomocí funkce *Interpolate* lze proložit křivku libovolnými body v prostoru. Na zkušebním modelu byla interpolační křivka vytvořena pro každou sloupovou větev a proložena body v

úrovni každé stropní desky viz Obrázek 5.22. Jednotlivé body na každé desce (modré kuličky) a jejich vodorovné souřadnice jsou proměnné hodnoty a vstupují jako jediné geny do evolučního řešiče.



Obrázek 5.22: Generování sloupů dle interpolační křivky

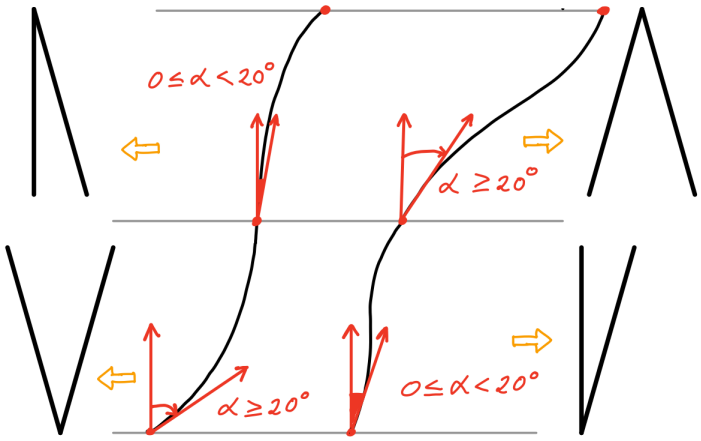
Vodící křivka kvůli své křivosti a proměnlivému tvaru mezi jednotlivými podlažími nabízí velký počet potenciálních parametrů, dle kterých lze sestavit algoritmus pro generování sloupů. Algoritmus je sestaven obdobným principem jako v sekci 5.5.2, kde každá větev začíná od nejníže položeného sloupu. Rozdílový je nyní fakt, že se sloupy generují pomocí vlastností křivky znázorněných na Obrázku 5.23.



Obrázek 5.23: Vlastnosti vodící křivky

Pro každou sloupovou větev je v každém podlaží vytvořeno pět parametrů, které vychází z vlastností křivky na daném úseku. Jednotlivé parametry byly vytvořeny tak, aby se výsledný sloupový prvek co nejvíce podobal vlastnostem křivky v daném podlaží a výběr dle vodící křivky byl logický. Zároveň byla snaha jednotlivé parametry udělat co nejméně vzájemně závislé, aby bylo možné dosáhnout co nejrozmanitějšího výběru.

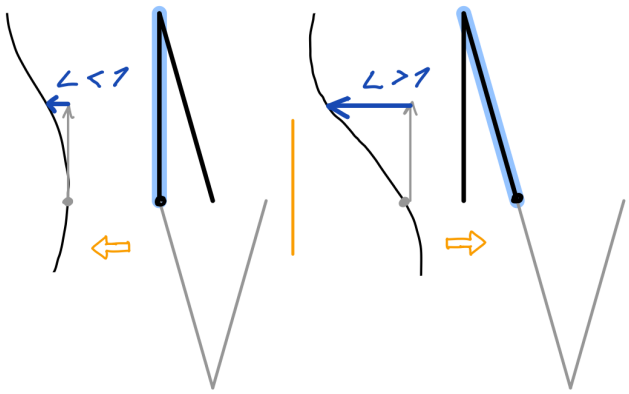
První parametr (na Obrázku 5.23 označen číslem 1) zkoumá konvexnost nebo konkávnost křivky z pohledu pozorovatele, který je umístěn v polovině výšky podlaží svisle od průsečíku s deskou. Posuzované místo je znázorněno fialovou šipkou. Toto kritérium slouží pro určení, zda typ sloupu bude tvaru A nebo tvaru V. Je-li křivka od pozorovatele prohnuta konkávně, algoritmus zvolí tvar sloupu V. Naopak, pokud je křivka prohnuta konvexně, je zvolen sloup tvaru A. Na první pohled je přiřazení tvaru ke křivce opačné a nelogické, což je způsobeno vodorovným pohledem od pozorovatele. Pokud na křivku nahlížíme přirozeně, např. jako na graf funkce směrem zdola nahoru, tak je přiřazení tvaru sloupu ke konvexnosti nebo konkávnosti křivky smysluplné.



Obrázek 5.24: Druhý parametr křivky

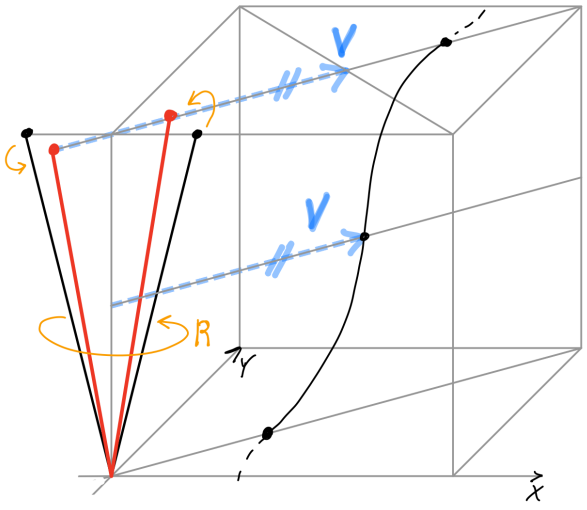
Druhý parametr upřesňuje finální tvar sloupového prvku. Prvním parametrem byl vybrán tvar sloupového prvku a druhým parametrem se určí, zda má být prvek symetrický nebo asymetrický. K rozhodování slouží úhel svírající dva vektory, který je měřen vždy od svislého vektoru k vektoru tangenty vodící křivky v místě průsečíku s deskou. Hodnota úhlu je vždy kladná a nabývá hodnot od 0° u svislého vektoru do zhruba 90° dle orientace vodící křivky. Jednoduché schéma je znázorněno na Obrázku 5.24. Rozhodujícím úhlem je 20 stupňů, tedy pokud je úhel menší než 20° je zvolen sloup asymetrický a pokud je úhel větší než 20° je zvolen sloup symetrický.

Třetí parametr je odvozený z parametru prvního tak, že používá jeho body k sestrojení vektoru, jehož délka je rozhodujícím kritériem pro výběr startovacího bodu u asymetrického sloupu tvaru A. Vektor je sestrojen od základny pozorovatele v polovině výšky podlaží k místě posuzovaném na křivce. Pokud je délka vektoru větší než 1 metr, tak algoritmus zvolí napojení šikmou částí sloupového prvku. Sloup se napojí svislou částí, pokud je délka vektoru menší než 1 metr. Zbývající typy sloupů posuzované nejsou, neboť druhý sloup tvaru A je symetrický, a tudíž jsou jeho variace řešeny rotací a sloupy tvaru V mají pouze jeden napojovací bod.



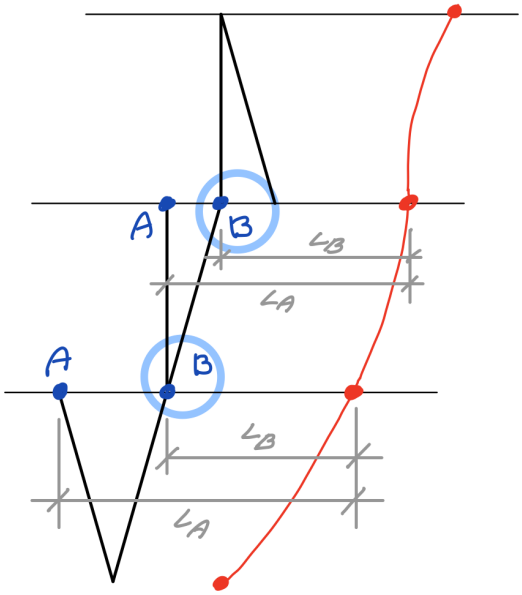
Obrázek 5.25: Třetí parametr křivky

Parametrem čtvrtým je rotace sloupů. Každý sloup v každém podlaží má nezávislou možnost rotace a otáčí se okolo svého startovacího bodu podle polohy vodící křivky v daném podlaží. Sloup se tedy vždy natočí do směru vektoru z parametru 1 nebo 3 viz Obrázek 5.23, který znázorňuje orientaci bodu v polovině křivky z pohledu od bodu startovacího.



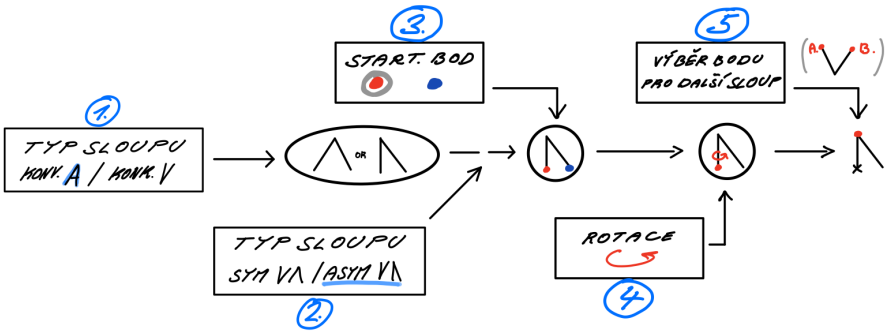
Obrázek 5.26: Parametr 4. - rotace sloupu

Posledním parametrem je potom vzdálenost mezi průsečíkem s podlažím a koncovými body. Tento parametr slouží k výběru koncového bodu u sloupu tvaru V, z kterého bude začínat sloup v následujícím podlaží. Algoritmus v každém podlaží vybere nejbližší bod sloupu k průsečíku vodící křivky a bod označí jako startovací pro další krok v algoritmu. Parametr rotace spolu s parametrem výběru bodu tak udávají celkovou orientaci sloupových větví.



Obrázek 5.27: Parametr 5. - výběr koncových bodů

Pomocí všech těchto parametrů algoritmus vygeneruje větev sloupů pro každou interpolační vodící křivku a jednotlivé sloupy lze následně ohodnotit dle optimalizačních kritérií, které vstupují jako fitness funkce do genetického algoritmu. Schéma algoritmu pro generování sloupů se všemi parametry je znázorněno na Obrázku 5.28.

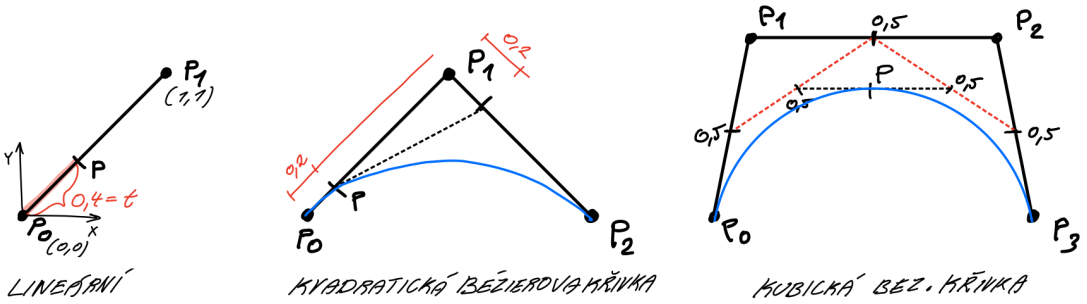


Obrázek 5.28: Schéma algoritmu pro generování sloupů

Bézierova křivka

Posledním vylepšením generování sloupů je úprava typu křivky. Na předchozích stránkách byl algoritmus řízen tzv. interpolační křivkou, tedy křivkou proloženou body v podlažích. Tyto body, jak už bylo popsáno, jsou proměnnou veličinou, která vstupuje do evolučního řešiče. Nevýhodou této metody je vazba genů (souřadnic bodů) s počtem jednotlivých stropních desek. S rostoucím počtem podlaží, totiž narůstá i velikost genomu a prodlužuje se čas výpočtu optimalizace.

Řešením bylo využít křivku, kterou lze sestrojit s menším počtem parametrů a křivka si stále zachová velkou variabilitu tvaru a křivosti. Po testování několika parametrický křivek byla zvolena křivka Bézierova. Tato křivka byla vytvořena francouzským inženýrem Pierrem Bézierem (* 1910, † 1999), který pracoval jako konstruktér pro automobilku Renault. Pomocí jeho metody bylo možné matematicky popsat křivky různých tvarů, které se využívaly zejména pro návrh karoserií osobních automobilů. Z Bézierovy křivky se postupně vyvinuly další typy křivek, které jsou spolu s původní křivkou hojně využívány například v počítačové grafice, počítačových hrách, CAD aplikacích apod. [19]



Obrázek 5.29: Příklady Bézierovy křivky

Bézierova křivka se vytváří pomocí řídícího polygonu a bodů na něm ležících. Na Obrázku 5.29 lze vidět základní tři typy Bézierových křivek. Na lineární křivce lze jednoduše vysvětlit základní princip, který je aplikovatelný na všechny typy. Dva body v prostoru nazvané P0 a P1 spojíme úsečkou a definujeme bod P, jehož polohu určuje tzv. parametr „t“, který nabývá hodnot od 0 v bodě P0 do 1 v bodě P1. Na lineárním typu vytvoří bod P přímku z bodu P0 do bodu P1. Jedná se o funkci lineární interpolace, která se dá zapsat jako:

$$P(t) = (1 - t) \cdot P_0 + t \cdot P_1, \quad 0 \leq t \leq 1$$

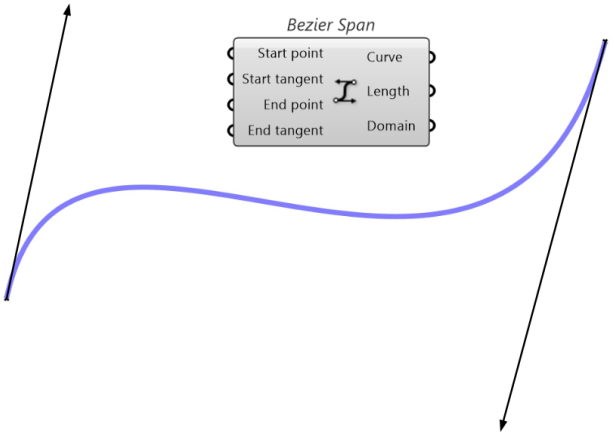
Kvadratickou křivku tvoří dvě úsečky mezi třemi body. Na dvou úsečkách jsou vytvořeny body P', které sdílí jeden parametr „t“. Nyní je nutné vytvořit další úsečku mezi dvěma body P', na které je opět vytvořen bod P se shodným parametrem „t“. Křivka vzniká pohybem bodů P' a P dle parametru „t“ od 0 do 1, kde trajektorie bodu P vytvoří finální Bézierovu křivku. Funkce P(t) se dá zapsat jako:

$$P(t) = (1-t)^2 \cdot P_0 + 2 \cdot (1-t) \cdot t \cdot P_1 + t^2 \cdot P_2, \quad 0 \leq t \leq 1$$

Kubická Bézierova křivka je ze všech typů nejrozšířenější a nejvíce používaná v již zmiňovaných odvětvích. Kubická křivka se vytváří pomocí řídícího polygonu složeného ze tří základních úseček. Na každé úsečce jsou vytvořeny tři body P", jejichž spojením vzniknou dvě úsečky s body P', které se znovu spojí a vytvoří poslední úsečku s bodem P. Finální křivku opět popisuje dráha funkce P(t), která je formulovaná jako:

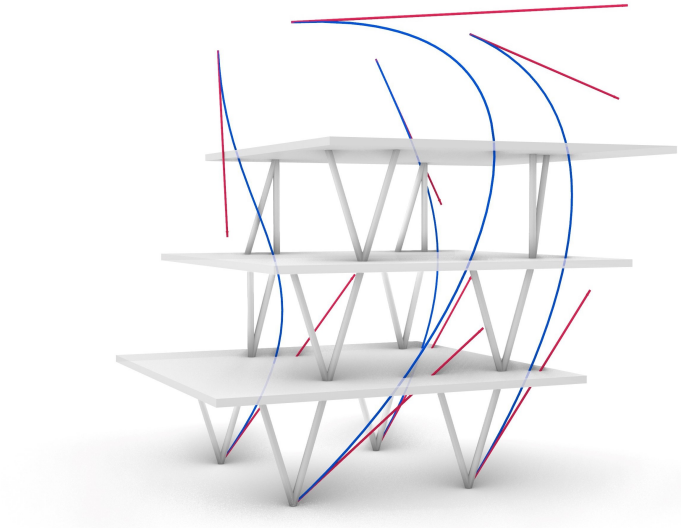
$$P(t) = (1-t)^3 \cdot P_0 + 3 \cdot (1-t)^2 \cdot t \cdot P_1 + 3 \cdot (1-t) \cdot t^2 \cdot P_2 + t^3 \cdot P_3, \quad 0 \leq t \leq 1$$

V programech a aplikacích je pro přehlednost často kubický tvar Bézierovy křivky zobrazován bez prostřední úsečky a dvě krajní úsečky mohou být nahrazeny vektory. Podobně je tomu i v prostředí Grasshopper, kde se dá Bézierova křivka vytvořit pomocí komponenty *Bezier Span*. Vstupními daty jsou: počáteční bod, počáteční tangenta, konečný bod a tangenta v konečném bodu. Tangenty si lze představit jako zmiňované krajní úsečky a mezilehlá prostřední úsečka by pak byla spojnice mezi dvěma šipkami na Obrázku 5.30.



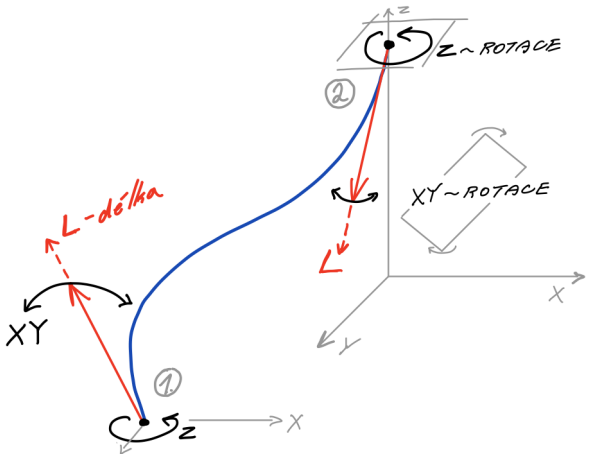
Obrázek 5.30: Bézierova křivka v prostředí Grasshopper

Pro generování sloupů je využito Bézierových křivek, kde každá křivka má počáteční bod v základně každé sloupové větve, která je v reálné konstrukci tvořena v místě základu nebo v místě hlavy sloupu z podzemních garáží. Koncový bod má křivka o jedno imaginární podlaží výše. Účelem toho je, aby souřadnice průsečíku křivky a poslední stropní desky byly proměnné a poslední sloup byl tak stále variabilní.



Obrázek 5.31: Generování sloupů dle Bézierovy křivky

Tvorba řídícího polygonu v prostoru je závislá na třech parametrech u počátečního bodu a na třech parametrech u koncového bodu křivky. Prvním z parametrů je rotace řídících úseček kolem osy Z, čímž se určuje orientace křivky v horizontální rovině. Druhým parametrem je vertikální pohyb řídících úseček, který je pojmenován rotace okolo os X,Y. Posledním parametrem je délka úsečky. Celkově je tedy jedna vodící křivka reprezentována šesti nezávislými parametry. Parametry jsou znázorněné na Obrázku 5.32.



Obrázek 5.32: Schéma tvorby Bézierovy křivky v prostoru

Kapitola 6

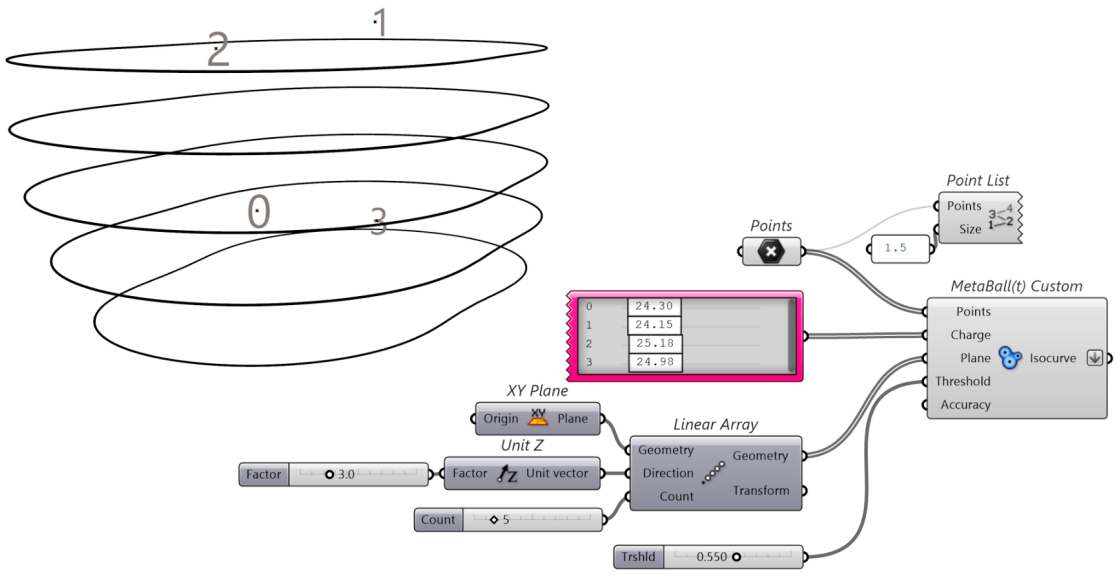
Praktická aplikace algoritmu

V poslední kapitole bude popsána praktická aplikace algoritmu na model budovy, který vznikl s využitím funkcí parametrického modelování. Cílem bylo jednoduše vytvořit model, který má jasně definované podlaží a stropní desky. Pro tento účel bylo využito komponenty *Metaball*, pomocí které lze generovat organické tvary v podobě uzavřených křivek. Ty lze následně převést na plochy a využít jako zmiňované stropní desky. Na Obrázku 6.1 je příklad několika takto vytvořených modelů vhodných k využití algoritmu pro generování sloupů. Z důvodu krátkého výpočetního času při optimalizaci a plynulého fungování algoritmu byl pro aplikaci algoritmu upřednostněn model méně rozměrný oproti vysokým nebo či rozlehlým modelům. Algoritmus je však univerzální a lze použít na jakýkoliv tvar budovy, jelikož vyžaduje pouze stropní desky a počáteční body jako vstupní data.

Obrázek 6.1: Modely vytvořené pomocí komponenty *Metaball*

6.1 Tvorba modelu

Při tvorbě konstrukce bylo vhodné zakomponovat vlastnosti zkušebních modelů z předchozí kapitoly. Odstupňované desky jednotlivých podlaží jsou vytvořeny tak, že na jedné straně jsou mírně vykonzolované a na druhé straně desky vytváří terasovitý tvar budovy. Sloupové větve tak mohou reagovat na změnu geometrie a přizpůsobit se jí. Samotné základní vytváření modelu je jednoduché a jde pouze o hledání vhodné polohy bodů, okolo kterých se vytváří kulový tvar *Metaballu*. Při přibližování bodů se kulové tvary začnou spojovat a vytváří tak zajímavé a velmi neotřelé organické tvary. Model budovy pro účely optimalizaci byl vytvořen splynutím dvou dvojic bodů do tvaru oválu a svým tvarem je úmyslně jeden z méně výrazných *Metaballů*.



Obrázek 6.2: Ukázka tvorby modelu pomocí komponenty *Metaball(t) Custom*

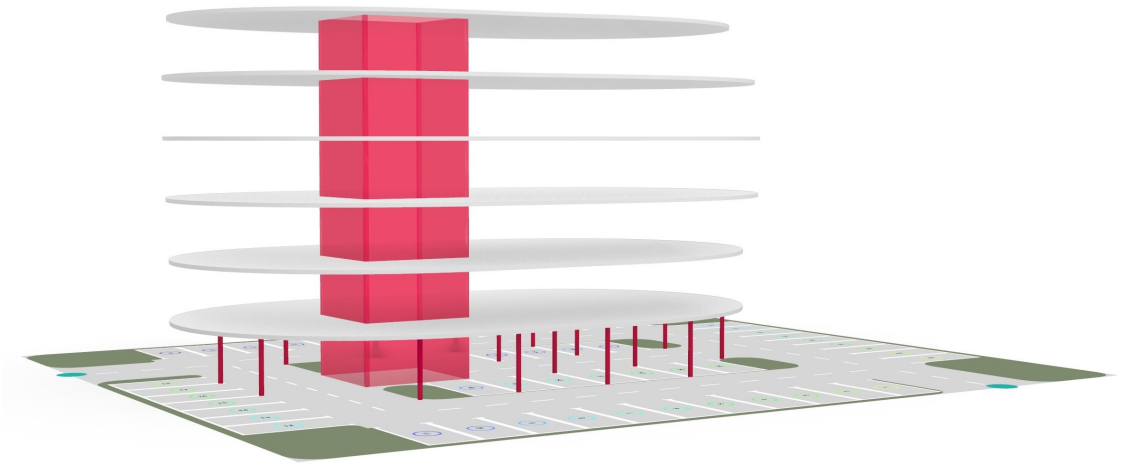
Na Obrázku 6.2 je ukázána tvorba modelu pomocí komponenty *Metaball(t) Custom*, která umožňuje podrobnější nastavení *Metaballu*. Základním nastavením je definování roviny vytváření, odstupu křivek (výšky podlaží) a počtu křivek (počtu podlaží). Následně je nutné určit body, které budou vytvářet jednotlivé *Metabally* (kulové tvary). Body lze definovat více způsoby, ale způsob použitý v DP je určení bodů v grafickém prostředí Rhino (na Obrázku 6.2 vlevo jsou body označené číslicemi). Jak už bylo zmíněno, úpravou polohy bodů, v tomto případě jejich přiblížením, došlo ke sloučení do jednoho oválného tvaru *Metaballu*.

6.1.1 Počáteční podmínky

V podsekci počáteční podmínky budou popsány dodatečné úpravy a podmínky, které dodají úloze optimalizace sloupů širší rozměr. První úpravou konstrukce je přidání ztužujícího jádra do budovy, které má za úkol budovu ztužit zejména v horizontálním směru a částečně tak kompenzovat absenci nosných stěnových prvků v budově. Další funkcí jádra je samozřejmě zprostředkování vertikální komunikace v budově pomocí schodiště a výtahu a jejich oddělení od volných prostor v objektu.

Parkoviště

Další důležitou podmínkou je stanovení počátečních bodů pro sloupové prvky. Aby určení počátečních bodů bylo smysluplné a konstrukce nabývala podobností s praxí, bylo tak navrženo podzemní parkoviště pod objektem, které udává jistý konstrukční rastr.



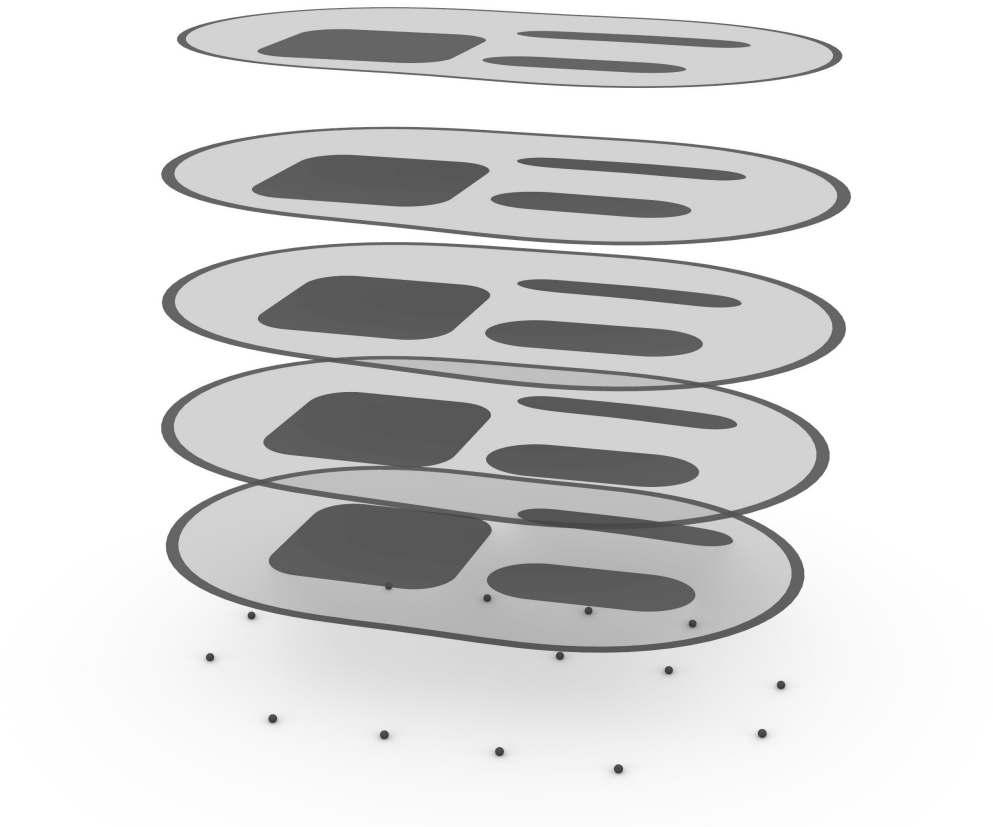
Obrázek 6.3: Model budovy se ztužujícím jádrem a rastrem parkoviště

Podzemní parkoviště bylo vytvořeno pomocí rozšiřujícího balíčku Parking Solver [20], což je nástroj vytvořený k rychlému a automatickému navržení parkoviště včetně parkovacích stáních a jízdních pruhů. S využitím tohoto nástroje byly navrženy rovné sloupy do podzemního parkoviště v rovnoměrném rastru ob každé jedno parkovací místo s umístěním na oba konce parkovacího místa. Jedno pole rastru zabírá tedy plochu $5x5m$ na parkovacích stáních nebo $5x6m$ v místě komunikace.

Chráněné prostory

Mezi počáteční podmínky algoritmu patří také omezení polohy sloupových prvků. Tato podmínka se přímo projevuje i v kritériích genetického algoritmu jako konfliktní kritérium, tedy kritérium, které minimalizuje polohu sloupových prvků v tzv. chráněných zónách. Tyto zóny jsou v praxi volné prostory v budově, kde není vhodné mít sloupy zejména z architektonického hlediska. Může se jednat o volná átria, velké zasedací místnosti apod. V budově se nachází tři takové prostory různé velikosti a funkce. První je chráněný prostor okolo schodišťového jádra určený pro hlavní chodbu. Ostatní dva chráněné prostory jsou pak ve volném prostoru a jsou určeny pro zasedací místnosti nebo jiné volné prostory.

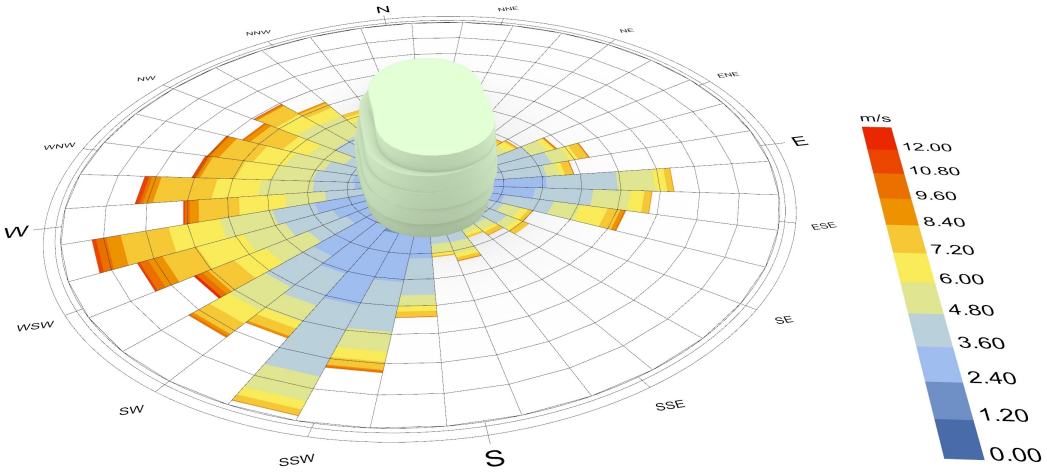
Algoritmus pro generování sloupů je nastavený tak, že do chráněných prostor sloup může zasahovat, ale vždy pouze slepou větví sloupového prvku. To znamená, že sloup z chráněné oblasti nemůže vzniknout, a proto je v takovém případě sloupová větev ukončena. Takovou ukončenou větev by genetický algoritmus měl vyhodnotit jako nevhodnou z kritéria statického a měla by zaniknout v průběhu evoluce.



Obrázek 6.4: Návrhový prostor pro sloupové prvky s vyznačenými chráněnými oblastmi

6.2 Analýza modelu

Tato sekce bude zejména věnována environmentální analýze budovy pomocí rozšiřujícího balíčku Ladybug Tools. Analýza prostředí byla provedena pomocí sady Butterfly, která je součástí balíčku Ladybug Tools, a slouží k provádění CFD analýz proudění (Computional Fluid Dynamics). Sady Butterfly bylo využito za účelem představy o působení větru na konstrukce nepravidelného či neobvyklého tvaru, a také za účelem získání dat pro vodorovné zatížení modelu.



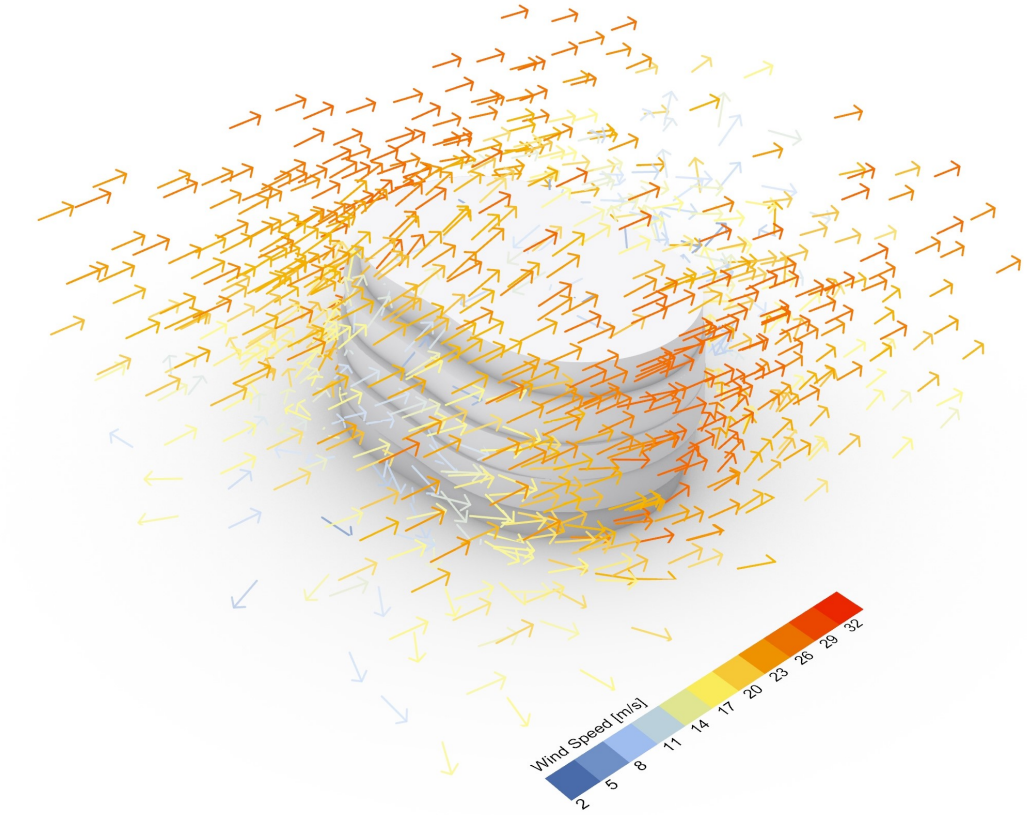
Obrázek 6.5: Větrná růžice z programu Ladybug

Prvně je nutné získat data o klimatu v určeném místě stavby. Pro tento účel bylo využito programu Ladybug, který umožňuje zvolit z mapy meteorologickou stanici a využít její data. Data jsou z meteorologické stanice Praha - Karlov v Klementinu. Základní sada Ladybug pak byla ještě využita k vytvoření větrné růžice okolo modelu zkoumané budovy. Větrná růžice znázorňuje četnost a rychlost větru v průběhu kalendářního roku. Z větrné růžice byl také určen západní směr větru, jako převládající směr s nejvyššími rychlostmi, pro analýzu proudění v programu Butterfly.

6.2.1 CFD simulace větru

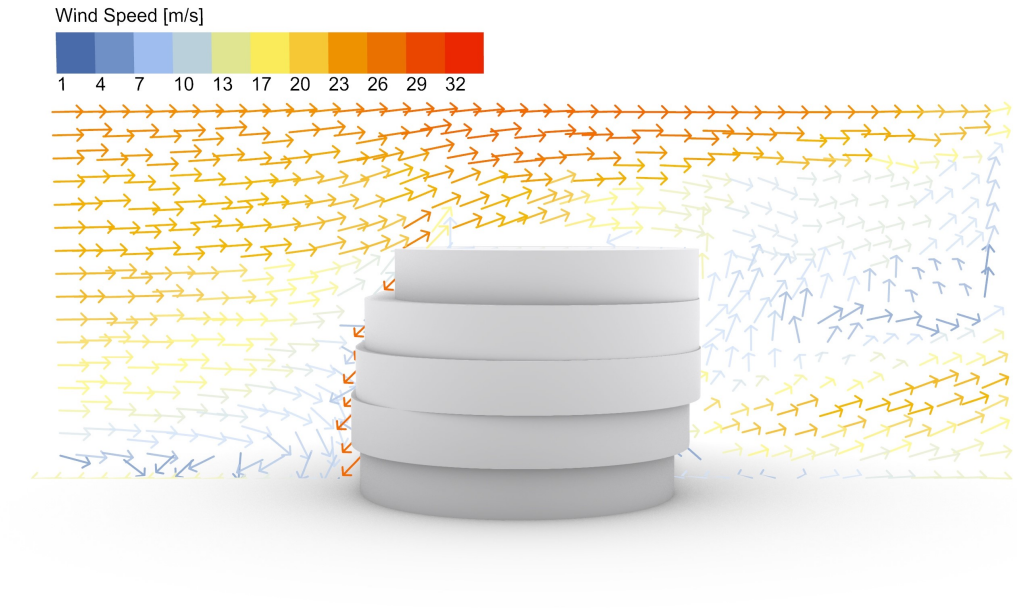
Pro CFD simulaci větru je nutné model budovy umístit do virtuálního větrného tunelu a nastavit jeho parametry, jako např. rozměry, rychlost a směr větru, úroveň terénu aj. Výchozí rychlost větru byla stanovena na vyšší hodnotu, než je v datech zmíněné meteorologické stanice, a to na $25 m/s$, což je hodnota, která je určena pro většinu území ČR dle mapy větrných oblastí. Referenční výška pro rychlost větru

je nastavena na výchozí hodnotu $10m$ a hrubost terénu na hodnotu 6 dle Davenportovo stupnice, kterou Butterfly využívá. Následuje zasíťování větrného tunelu včetně umístěného modelu ve dvou krocích, kdy v prvním kroku je prováděno hrubé síťování a v druhém kroku je síť vyhlazována v okolí modelu. Poslední částí je nastavení a výpočet simulace, ve které si uživatel vytvoří vlastní síť, na které chce zobrazit výsledky, jako např. rychlost větru či tlak a sání větru. Výsledky CFD simulace byly také testovány a porovnávány s příkladem kupole s kruhovou základnou z normy *ČSN EN 1991-1-4 Zatížení větrem* a s ostatními dostupnými příklady pro věrohodnost výsledků.



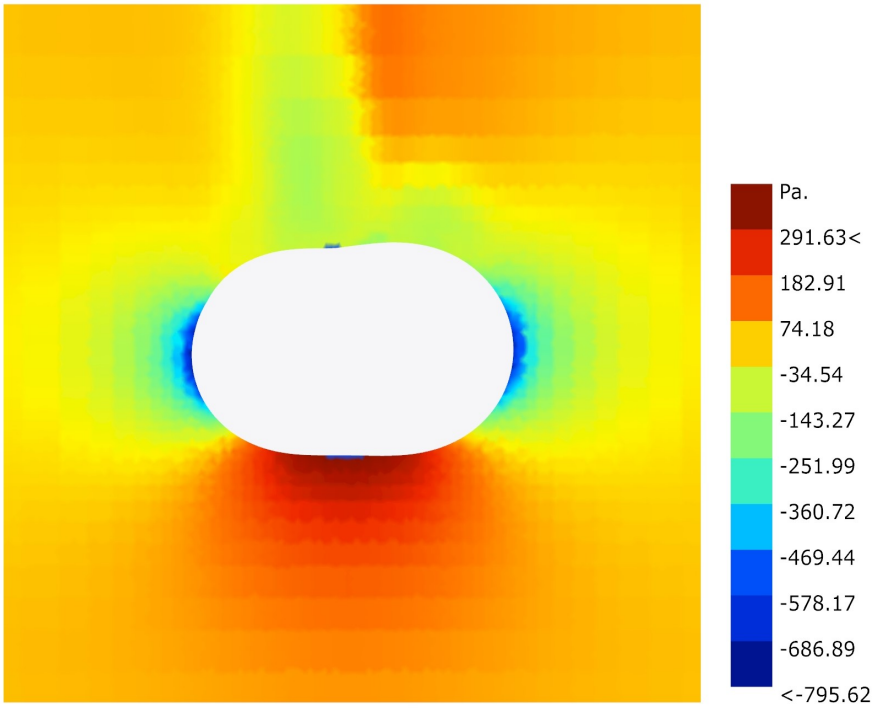
Obrázek 6.6: Zobrazení rychlosti a směru větru v okolí modelu

Na Obrázku 6.6 je zobrazen výstup z CFD analýzy ve formě rychlosti větru. Rychlost byla analyzována v bodech umístěných v prostoru kolem modelu, aby bylo zobrazeno obtékání budovy větrem. K obrázku je přidána legenda rychlostí větru v jednotkách m/s, dle které lze vyčíst největší rychlosti větru v místech příčných zaoblení budovy, kde lze také očekávat největší hodnoty sání větru. Naopak nejmenší hodnoty rychlosti větru jsou v nižších polohách a v místech přímého tlaku větru (viz Obrázek 6.7).



Obrázek 6.7: Zobrazení rychlosti větru na svislé rovině

Zásadním výstupem z programu Butterfly pro určení zatížení je tlak větru. Výsledky tlaku větru se zobrazují na síti v požadované rovině, kterou lze vytvořit pomocí bodů používaných na zobrazení výsledků rychlosti větru. Program Butterfly pak na vytvořené síti vypočítá hodnoty tlaku větru, kde hodnoty záporné představují sání větru a hodnoty kladné tlak větru (viz Obrázek 6.8).



Obrázek 6.8: Zobrazení tlaku a sání větru z půdorysného pohledu

6.2.2 Statická analýza

Ze zobrazení působení tlaku a sání větru na konstrukci byly stanoveny oblasti kladného a záporného tlaku spolu s jejich hodnotami, dle kterých bylo namodelováno vodorovné zatížení na jednotlivé desky. Zatížení na konstrukci bylo aproximováno a rozděleno do 8 oblastí, které korespondují s Obrázkem 6.8 a hodnoty byly přenásobeny bezpečnostním součinitel o hodnotě 1.5.

Další nastavení statické analýzy vychází ze sekce Analýza modelu 5.2, kde jsou jednotlivé kroky podrobně popsány na zkušebním modelu. Sloupové prvky jsou nastaveny jako ocelové trubky s průměrem 194 mm a tloušťkou stěny 16 mm stejně jako v sekci 5.2.2, kde je popsáno i ověření únosnosti s programem Karamba3D. Deska je definována jako železobetonová s betonem třídy C30/37 a tloušťkou 20 cm. Zatížení je doplněno o svislé zatížení vlastní tíhou betonové desky a o svislé zatížení o velikosti $5kN/m^2$. Dále jsou doplněny pevné podpory pro sloupové větve do míst určených z počátečních podmínek. Typy výsledků neoptimalizovaného modelu zobrazené níže jsou zároveň používaná kritéria, v optimalizaci a jsou zde uvedeny pro porovnání.

Průhyb [cm]	Využití desek [%]
8.52	110.34

6.3 Optimalizace modelu

K optimalizaci modelu je využíván vyvinutý algoritmus pro generování sloupů spolu s programem Karamba3D a evolučním řešičem Wallacei X. Vstupními údaji do Wallacei X je genotyp, který byl získaný z algoritmu pro generování sloupů spolu s fenotypem, který reprezentuje geometrii sloupů. Dalším vstupním údajem jsou kritéria optimalizace získané ze statické analýzy Karamba3D a kritérium penalizace, které představovaly průsečíky s chráněným prostorem. Kritéria jsou přehledně vypsána níže:

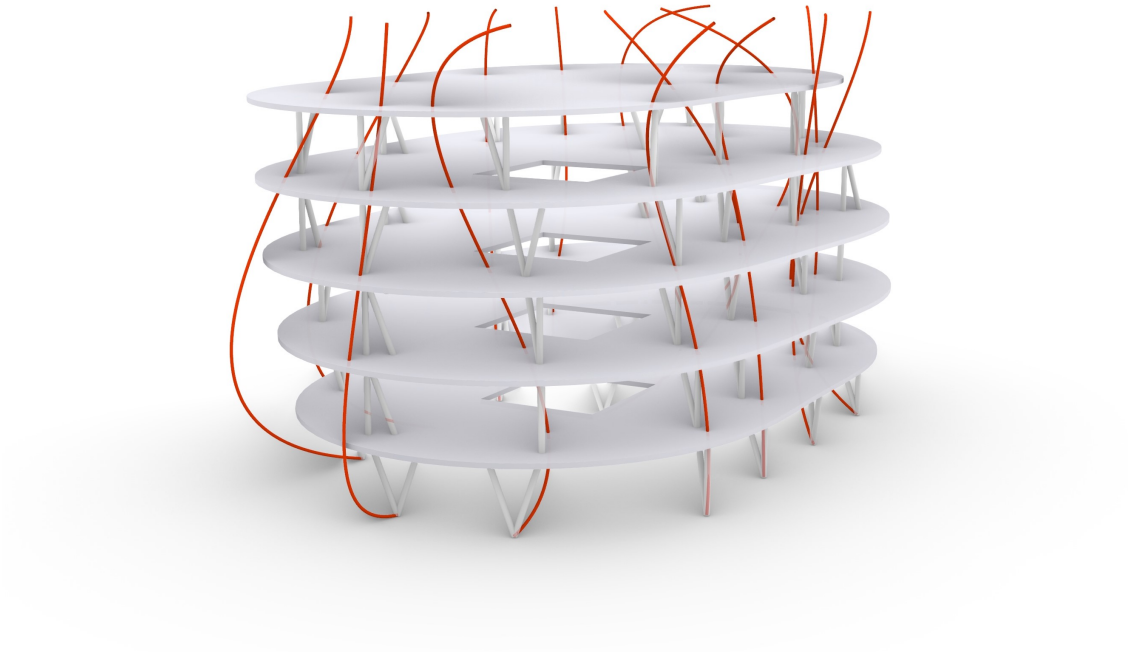
- K1 - Deformace (součet maximálních svislých a vodorovných deformací)
- K2 - Využití desek (max. využití desek zprůměrované do jednoho kritéria)
- K3 - Konflikty (průměrná hodnota střetů sloupů s chráněným prostorem)

V kapitole vývoje algoritmu 5 bylo využíváno více kritérií, avšak po důkladném testování dosahovala optimalizace nejlepších výsledků s použitím kritérií výše vypsaných.

Populace	Velikost generace	80
	Počet generací	100
Parametry algoritmu	Pravděpodobnost křížení	0.9
	Pravděpodobnost mutace	0.1
	Index distribuce křížení	5
	Index distribuce mutace	5
	Náhodný seed	0
Parametry simulace	Počet sliderů s geny	84
	Počet proměnných hodnot (genů)	20958
	Počet kritérií	3
	Velikost prohledávaného prostoru	3.3e199

Tabulka 6.1: Nastavení genetického algoritmu Wallacei X

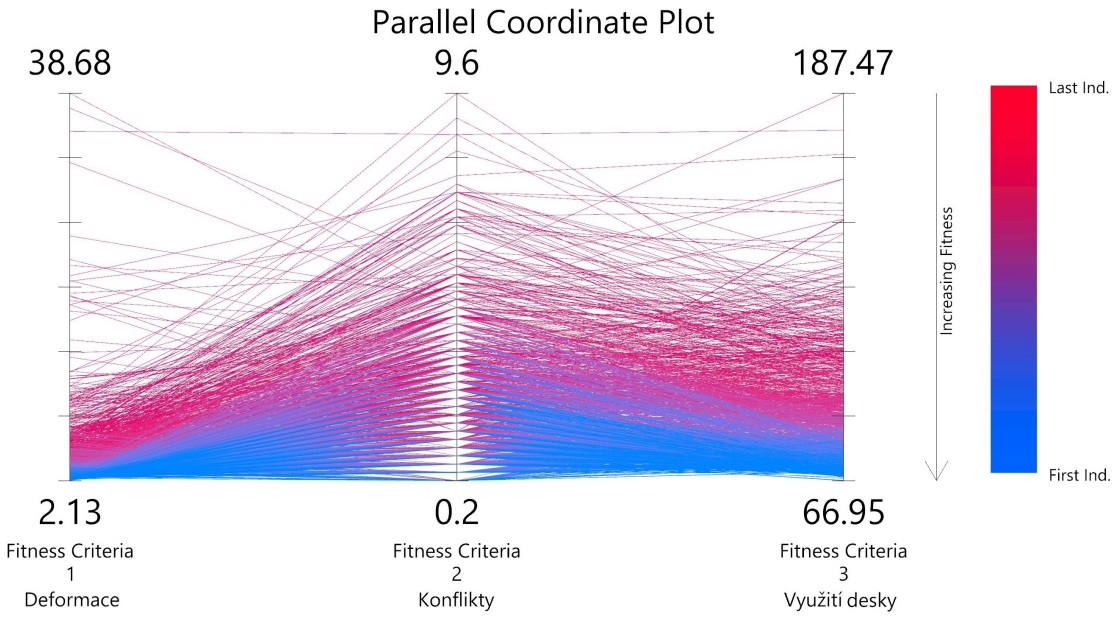
Genotyp se pro připomenutí skládá z parametrů křivky, které byly řešeny v sekci 5.5, podle kterých se generují sloupy ve čtyřech variantách. Varianty sloupů a řešení jejich napojení včetně výztuh proti protlačení je inspirováno návrhem budovy Active Energy Building popsaným v kapitole 4. Aplikace algoritmu generování sloupů se zobrazením vodících křivek je znázorněna na Obrázku 6.9 níže.



Obrázek 6.9: Zobrazení vodících křivek na modelu budovy

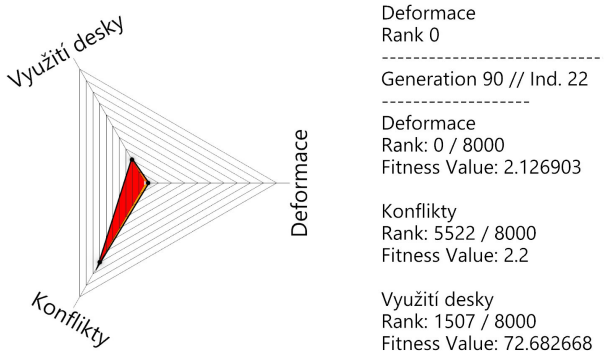
6.3.1 Výsledky optimalizace

Výsledky optimalizace lze zobrazit například na grafu PCP (Parallel Coordinate Plot) viz Obrázek 6.10, na kterém jsou znázorněna ohodnocená řešení genetického algoritmu. Každý jedinec je reprezentován jednou křivkou a nabývá hodnot fitness (zdatnosti) na svislých osách optimalizačních kritérií.

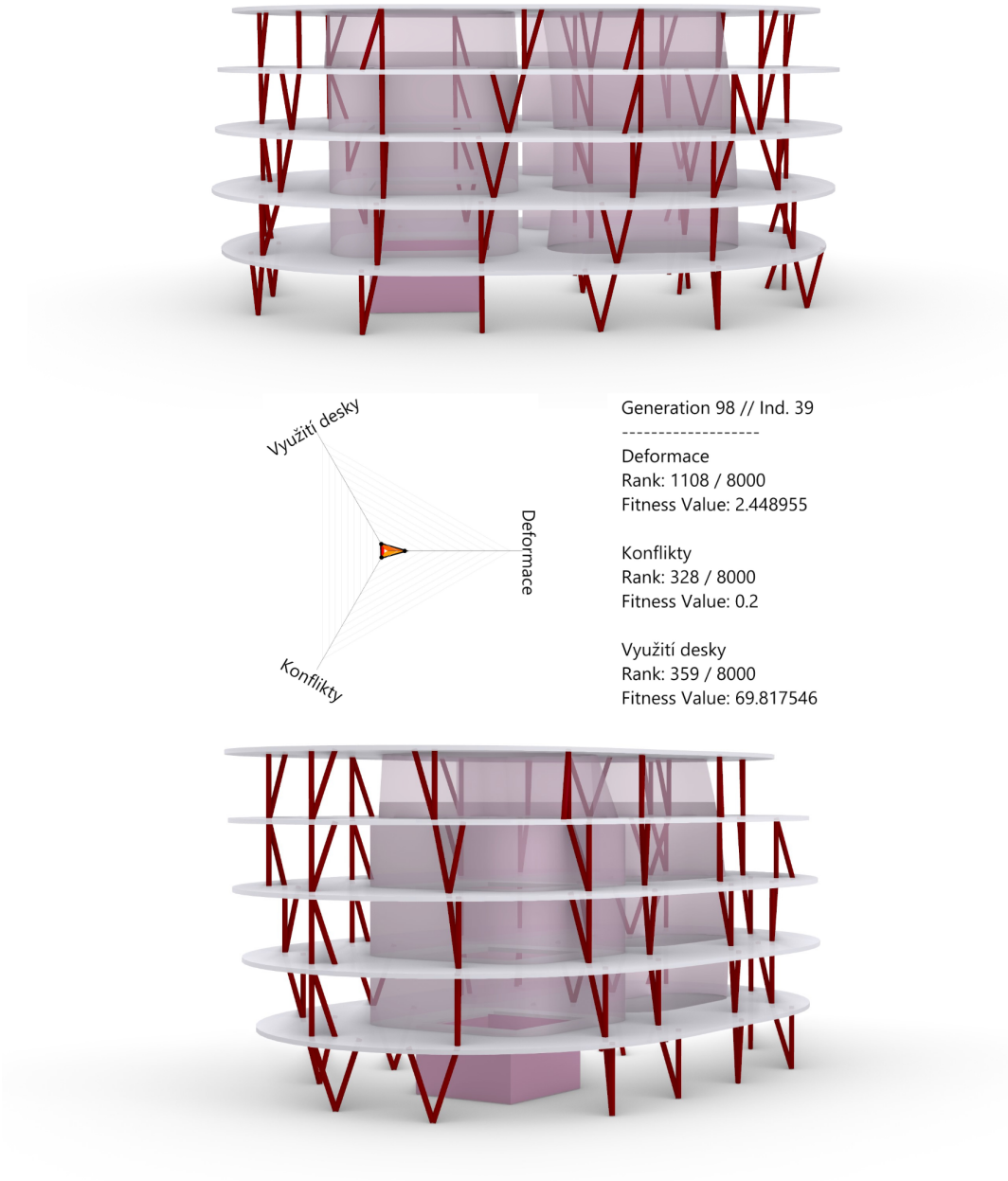


Obrázek 6.10: PCP (Parallel Coordinate Plot) optimalizace

Z grafu lze také vyčíst maximální a minimální hodnoty fitness a jelikož je snahou genetického algoritmus NSGA-II kritéria minimalizovat, tak si lze povšimnout husté koncentrace jedinců ve spodní části grafu v blízkosti nejlepších (minimálních) hodnot fitness.



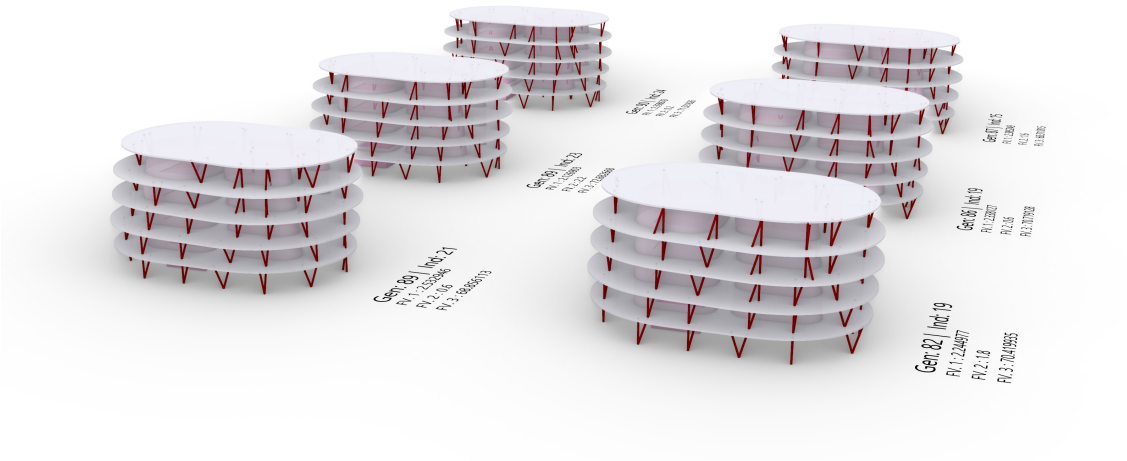
Obrázek 6.11: Řešení s minimální hodnotou deformace na paprskovém grafu



Obrázek 6.12: Řešení s minimálním průměrem fitness hodnot

Na obrázku výše je zobrazené nejlepší řešení vyhodnocené dle metody průměru fitness hodnot z grafu PCP, na kterém je zároveň zobrazena jako průměrně nejníže položená křivka. To lze také vyčíst z paprskového grafu, kde jedinec sice neexceluje ani v jednom z posuzovaných kritérií, ale průměrně dosahuje lepších hodnot než ostatní jedinci. Nezbytnou metodou pro vyhodnocení vícekriteriální optimalizace je Pareto množina nedominovaných řešení, která se v tomto případě skládá ze 47 jedinců z několika posledních generací. Pareto fronta by v praxi mohla sloužit jako koš naplněný nejlepšími řešeními, z kterých by si např. architekt či investor mohl

vybírat dle jejich požadavků s jistotou, že vybrané řešení bude např. vždy staticky efektivní. Níže proto bude znázorněno, jak by mohl vypadat výběr jedinců z Pareto množiny optimálních řešení. Na Obrázku 6.13 je tento výběr zúžený na 6 jedinců z Pareto fronty z důvodu zachování přehlednosti.



Obrázek 6.13: Zobrazení několika jedinců z Pareto množiny řešení

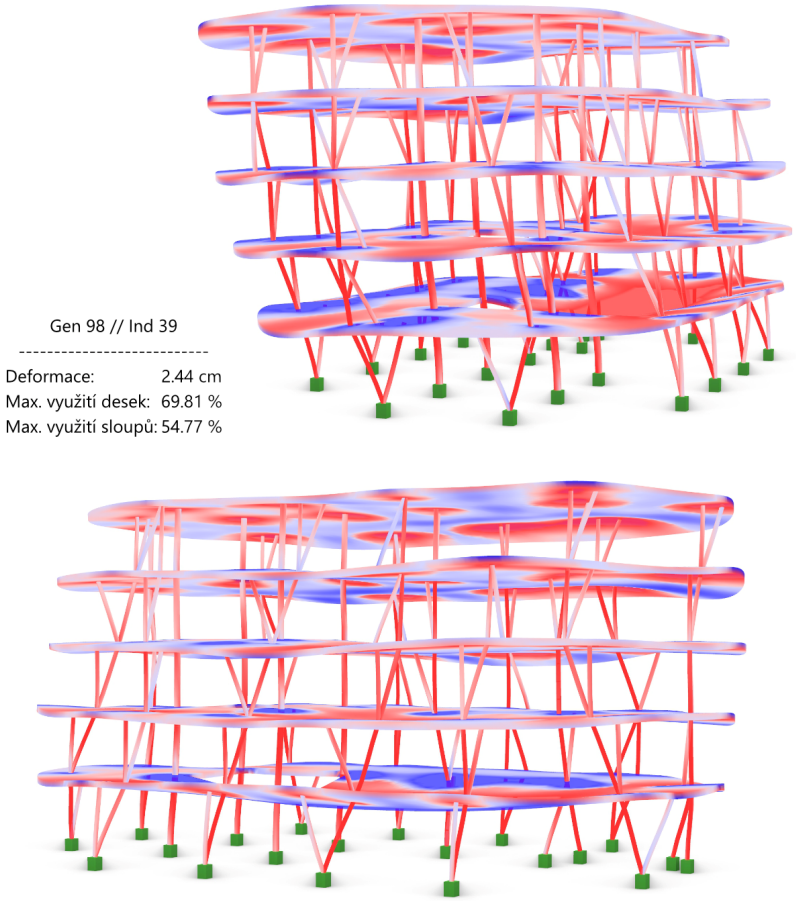
Ačkoliv jsou jednotliví jedinci spolu s generací a fitness hodnotami zobrazeny u vizuálního výběru z Pareto množiny, mohou být tato data vypsána i do přehledné tabulky viz níže, kde jsou zvýrazněny nejlepší fitness hodnoty. Tabulka byla zredukována na 10 jedinců, kteří byli vybrání pomocí shlukovací metody K-Means v řešiči Wallacei X, kde tito jedinci tvoří tzv. středy shluků.

Řešení		Kritérium		
Generace	Jedinec	Deformace[cm]	Konflikty[]	Využití desek[%]
82	19	2.25	1.8	70.42
87	19	2.20	1.2	70.75
89	21	2.53	0.6	68.86
89	23	**2.13**	2.2	72.68
91	26	2.28	1.2	69.73
95	32	2.41	1	**68.59**
95	34	2.36	0.4	70.30
98	39	2.44	**0.2**	69.81
98	41	2.35	1.8	68.93
99	35	2.32	2.2	68.84

Tabulka 6.2: Výběr řešení z Pareto fronty

Zhodnocení optimalizace

Optimalizaci sloupových prvků hodnotím jako úspěšnou, a ačkoliv bych na první pohled polohu určitých sloupy pozměnil, tak jsem při ručních změnách polohy zjistil, že ve většině případů dochází ruční úpravou ke zhoršení vlastností konstrukce a hodnot fitness. Velkou výhodu podobné vícekriteriální optimalizace vidím v zapojení lidského faktoru do výběru finální konstrukce z několika optimálních řešení, což může být např. při spolupráci několika členů u návrhu stavby velmi přínosné.



Obrázek 6.14: Zobrazení napětí na deformované konstrukci

Výše vybrané řešení nabývá nejnižší hodnoty konfliktů, což znamená, že jedinec má nejmenší počet průsečíků sloupů s chráněným prostorem, a může tak být vhodným řešením k výběru z architektonického hlediska. Ze statického pohledu dochází k deformaci 2.44 cm na spodní desce, kde se z důvodu aplikace chráněného prostoru nemohou generovat sloupy dle rastru parkoviště, čímž se navyšuje rozpětí desky na 10 m, které variabilní sloupy postupně zmenšují. Zároveň je vhodné podotknout, že se na obrázku jedná o statickou analýzu v rané fázi projektu, která má hlavně sloužit k představě chování konstrukce a k prvotním návrhů a nenahrazuje tak podrobnější statickou analýzu, která by byla provedena v pozdější fázi návrhu.

Kapitola 7

Závěr

Cílem diplomové práce bylo využít parametrické modelování spolu s genetickými algoritmy k optimalizaci polohy a tvaru sloupových prvků. Tento cíl byl splněn a optimalizace sloupových prvků byla provedena více způsoby. V druhé kapitole byly využity obecné metody k optimalizaci topologie. První topologickou metodou byla optimalizace rozmístění pomocí BESO algoritmu, který byl použit ve formě rozšíření Karamba3D. Tato forma se ukázala jako vhodná spíš pro prutové prvky v příhradové nebo roštové konstrukci. Druhá topologická optimalizace byla optimalizace tvaru pomocí metody SIMP. Metoda byla využívána pomocí rozšíření tOpos, ze kterého byly výstupy optimalizace velmi zajímavé a vhodné zejména pro výstavbu konstrukcí pomocí 3D tisku, nebo pro výrobu speciálně tvarovaného bednění. Diplomová práce byla od výběru témata inspirována návrhem sloupových prvků v budově Active Energy Building, kde byl navržen omezený počet sloupových prvků s účelem využití prefabrikace.

Metody návrhu omezeného počtu typů sloupů s možností prefabrikace jsem chtěl využít ve své diplomové práci, a proto jsem již v topologické optimalizaci nepokračoval a rozhodl se jít cestou vývoje vlastního algoritmu. Spolu s využitím evolučního řešiče Wallacei a statického programu Karamba3D jsem postupně vytvořil algoritmus, který lze aplikovat na různé tvary konstrukcí, což je provedeno v kapitole praktické aplikace algoritmu. Ve vývoji algoritmu by se dalo dále pokračovat a vytvořit např. možnost generovat další sloupové prvky ze slepých větví sloupů, které by se dokázaly větvit jako v případě topologické optimalizace tvaru. Využití algoritmu si lze představit buď v rané fázi návrhu, kde by byl součástí tzv. form findingu, nebo naopak v pozdějších fázích, kde by sloužil k finálnímu nalezení řešení.

Bibliografie

[1] D. Davis, *A History of Parametric*, srp. 2013. URL: https://www.danieldavis.com/a-history-of-parametric/.

[2] P. Block, M. DeJong a J. Ochsendorf, "As Hangs the Flexible Line: Equilibrium of Masonry Arches", *Nexus Network Journal*, roč. 8, s. 13–24, říj. 2006. DOI: 10.1007/s00004-006-0015-9.

[3] J. Frazer, "Parametric Computation: History and Future", *Architectural Design*, roč. 86, č. 2, s. 18–23, 2016. DOI: https://doi.org/10.1002/ad.2019.

[4] S. Zexin a H. Mei, "Robotic Form-Finding and Construction Based on the Architectural Projection Logic", *IOP Conference Series: Materials Science and Engineering*, roč. 216, s. 012 058, čvn. 2017. DOI: 10.1088/1757-899X/216/1/012058.

[5] C. Preisinger, "Linking Structure and Parametric Geometry", *Architectural Design*, roč. 83, č. 2, s. 110–113, 2013. DOI: https://doi.org/10.1002/ad.1564.

[6] K. Jebari, "Selection Methods for Genetic Algorithms", *International Journal of Emerging Sciences*, roč. 3, s. 333–344, pros. 2013.

[7] K. Deb, "Multiobjective Optimization Using Evolutionary Algorithms. Wiley, New York", in led. 2001.

[8] K. Deb, A. Pratap, S. Agarwal a T. Meyarivan, "A fast and elitist multiobjective genetic algorithm: NSGA-II", *Evolutionary Computation, IEEE Transactions on*, roč. 6, s. 182–197, květ. 2002. DOI: 10.1109/4235.996017.

[9] M. Makki, M. Showkatbakhsh a Y. Song. "Wallacei Primer 2.0". (2019), URL: https://www.wallacei.com/.

[10] V. Ullmann, *Topologie*. URL: https://astronuklfyzika.cz/Gravitace3-1.htm.

[11] I. Md Rian a M. Sassone, "Tree-inspired dendriforms and fractal-like branching structures in architecture: A brief historical overview", *Frontiers of Architectural Research*, roč. 3, zář. 2014. DOI: 10.1016/j.foar.2014.03.006.

[12] T. Mareš, *Základy konstrukční optimalizace*. 2006, s. 307, ISBN: 80-239-6508-5.

[13] S. Białkowski, "Structural Optimisation Methods as a New Toolset for Architects", srp. 2016. DOI: 10.52842/conf.ecaade.2016.2.255.

[14] M. Bruyneel a P. Duysinx, "Topology optimization with self-weight loading: (un-expected) problems and solutions", říj. 2001.

[15] *SIMP method for topology optimization*. URL: https://help.solidworks.com/2019/english/solidworks/cworks/c_simp_method_topology.htm.

[16] "Active Energy Building". (2016), URL: https://www.karamba3d.com/project/active-energy-building/ (cit. 01. 11. 2022).

[17] A. Falkeis. "Building Innovation for an Architecture in Motion", YouTube. (2019), URL: https://youtu.be/wMb0y1rGO08.

[18] R. Vierlinger, A. Hofmann a K. Bollinger, "Emergent Hybrid Prefab Structures in Dwellings", zář. 2013. DOI: 10.13140/RG.2.1.3351.8809.

[19] C. Rabut, "On Pierre Bézier's life and motivations", *Computer-Aided Design*, roč. 34, s. 493–510, čvn. 2002. DOI: 10.1016/S0010-4485(01)00121-X.

[20] C. Siebje, *Parking Solver*, čvn. 2022. URL: https://www.food4rhino.com/en/app/parking-solver.