Programski paket za robusno optimiranje topologije rešetkastih konstrukcija

Rožić, Mateja

SVEUČILIŠTE U ZAGREBU
FAKULTET STROJARSTVA I BRODOGRADNJE

DIPLOMSKI RAD

Mateja Rožić

Zagreb, 2016.

SVEUČILIŠTE U ZAGREBU
FAKULTET STROJARSTVA I BRODOGRADNJE

DIPLOMSKI RAD

Mentor:

Dr. sc. Andrej Jokić, dipl. ing.

Student:

Mateja Rožić

Zagreb, 2016.

Izjavljujem da sam ovaj rad izradila samostalno koristeći stečena znanja tijekom studija i navedenu literaturu.

Zahvaljujem se mentoru Dr. sc. Andreju Jokiću, dipl. ing za pruženu pomoć i savjete prilikom izrade završnog rada. Također mu se zahvaljujem i na pokazanoj inicijativi i spremnosti tijekom konzultacija.

Zahvaljujem se obitelji na pruženoj potpori tijekom školovanja.

Mateja Rožić

SVEUČILIŠTE U ZAGREBU
FAKULTET STROJARSTVA I BRODOGRADNJE
Središnje povjerenstvo za završne i diplomske ispite
Povjerenstvo za diplomske ispite studija strojarstva za smjerove:
procesno-energetski, konstrukcijski, brodostrojarski i inženjersko modeliranje i računalne simulacije

Sveučilište u Zagrebu Fakultet strojarstva i brodogradnje	
Datum	Prilog
Klasa:	
Ur.broj:	

DIPLOMSKI ZADATAK

Student: Mateja Rožić Mat. br.: 0035164922

Naslov rada na hrvatskom jeziku: **Programski paket za robusno optimiranje topologije rešetkastih konstrukcija**

Naslov rada na engleskom jeziku: **Software package for robust optimization of truss topology**

Opis zadatka:

U okviru diplomskog rada potrebno je osmisliti i razviti programski paket za robusno optimiranje topologije nosivih rešetkastih konstrukcija. Rad je potrebno temeljiti na najnovijim znanstvenim rezultatima u području robusnog optimiranja.

U okviru ovog zadatka potrebno je

1) Predstaviti prikladan pregled literature na temu optimiranja nosivih konstrukcija, s naglaskom na probleme robusnosti konstrukcije.
2) Matematički formulirati optimizacijski problem s krutosti konstrukcije kao funkcijom cilja, koji uključuje problem robusnosti konstrukcije na perturbacije od nominalnih opterećenja kao i na nepredvidiva opterećenja.
3) Izabrati prikladan programski jezik i postojeće računalne pakete (implementirane numeričke algoritme) prikladne za rješavanje postavljenog problema.
4) Razvijeni programski paket treba predati u elektronskom obliku uz izrađenu dokumentaciju za njegovo korištenje.
5) Na nekoliko odabranih primjera ilustrirati rad razvijenog programskog paketa. Na primjerima ilustrirati ostvarenu robusnost dizajniranih konstrukcija.
6) Dati prijedloge i predstaviti smjernice za inženjersku praksu oblikovanja robusnih nosivih konstrukcija.

U radu je potrebno navesti korištenu literaturu i eventualno dobivenu pomoć.

Zadatak zadan:	Rok predaje rada:	Predviđeni datumi obrane:
12. studenog 2015.	14. siječnja 2016.	20., 21. i 22. siječnja 2016.

Zadatak zadao:

Izv. prof. dr. sc. Andrej Jokić

Predsjednica Povjerenstva:

Prof. dr. sc. Tanja Jurčević Lulić

SADRŽAJ

POPIS SLIKA

POPIS TABLICA

POPIS OZNAKA

Oznaka	Jedinica	Opis
l	mm	početna duljina štapa
E	MPa	Youngov modul elastičnosti
σ	N/mm^2	naprezanje u štapu
ε	-	relativno produljenje štapa
A_{AB}	mm^2	poprečni presjek štapa AB
t_{AB}	mm^3	volumen štapa
F	N	sila u štapu
F_{RB}	N	Sila reakcije u točki B
β_{AB}	$\frac{\sqrt{N}}{mm^2}$	vektor u smjeru od čvora A prema čvoru B
E_p	J	potencijalna energija pohranjena u štapu
v	mm	pomaci čvorova
$A(t)$	$\frac{N}{m}$	matrica krutosti
$Compl_f(t)$	mm/N	podatljivost uslijed sile f
w	mm^3	ukupni dozvoljeni volumen konstrukcije
w^*	mm^3	ukupni optimalni volumen konstrukcije
f	N	vanjsko opterećenje,
$E_c(v)$	J	potencijalna energija pohranjena u konstrukciji kao rezultat pomaka *v*
V	-	linearni prostor virtualnih pomaka konstrukcije
τ	mm/N	podatljivost
Ƒ	N	nominalno opterećenje i nesigurnosti opterećenja svih čvorova.
I	-	jedinična matrica
Q	N	matrica opterećenja
r	mm	radijus kugle nepredvidivih opterećenja

SAŽETAK

U ovom diplomskom radu opisano je što je to linearno, a što semidefinitno programiranje, te su dani opći oblici njihovih zapisa. Rad se nastavlja matematičkom formulacijom problema topološkog optimiranja rešetkastih konstrukcija.

Nadalje, opisana je formulacija linearnog programiranja za optimiranje topologije rešetkastih konstrukcija kao i rješenja problema koji se se javili tijekom konstruiranja inicijalne konstrukcije („osnovne strukture“).

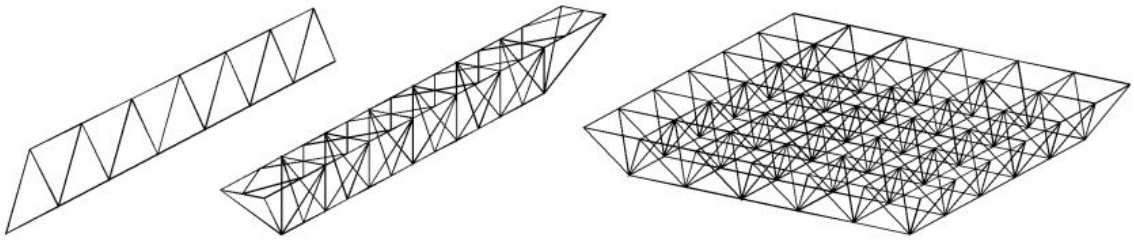
Rad se nastavlja ilustracijom potrebe za robusnosti konstrukcije na određenim primjerima. Nadalje, opisano je kako nominalni problem topološkog optimiranja rešetkastih konstrukcija reformulirati u obliku problema semidefinitnog optimiranja. U radu je definirano što se smatra robusnim optimiranjem rešetkastih konstrukcija, te je prikazana formulacija problema za robusno optimiranje rešetkastih konstrukcija.

Rad se nastavlja opisivanjem načina korištenja izrađenog programskog koda za robusno optimiranje rešetkastih konstrukcija.

Ključne riječi: robusno optimiranje, semidefinitno programiranje, rešetkaste konstrukcije, podatljivost.

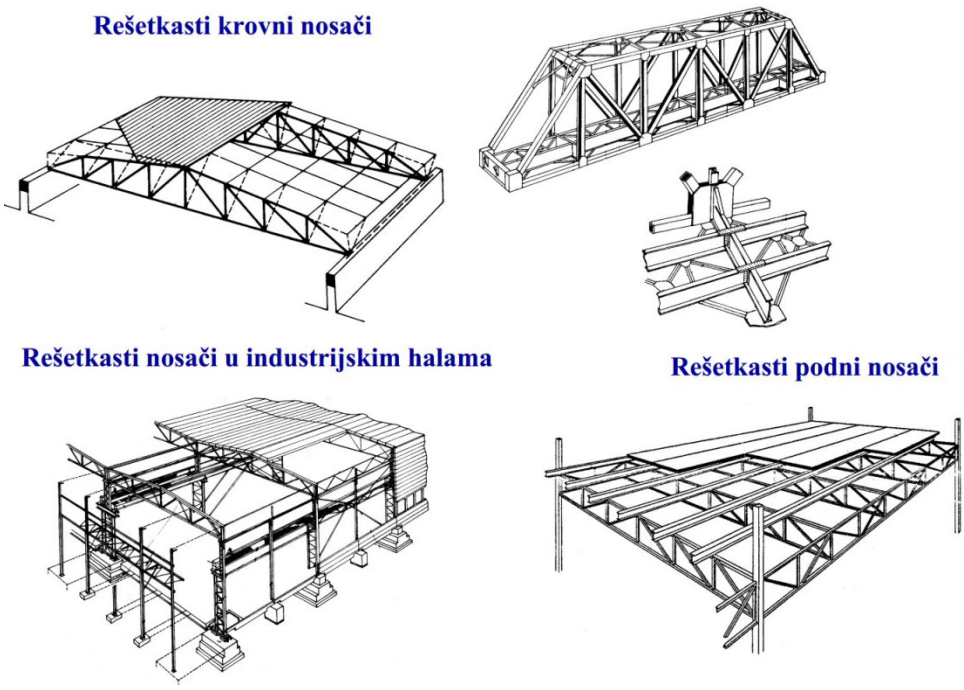
1. Uvod

Rešetka je mehanička konstrukcija koju čine elastični štapovi koji su međusobno povezani jedni s drugima u čvorovima. Prema obliku razlikuju se ravninske i prostorne rešetke [slika 1.1.].



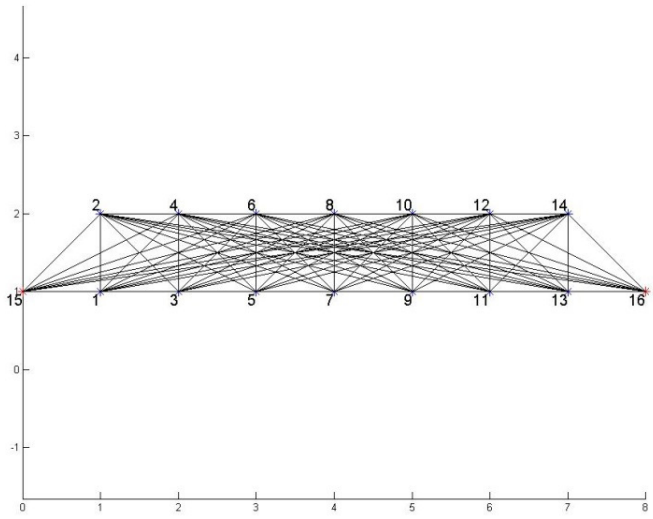
Slika 1.1. Ravninske i prostorne rešetkaste konstrukcije [1].

Primjeri rešetkastih konstrukcija su: željeznički mostovi, stupovi dalekovoda, rešetkasti krovni nosači, rešetkasti podni nosači i mnogi drugi. Neke od navedenih rešetkastih konstrukcija prikazane su na slici 1.2. Cilj programa predstavljenog u ovom radu je konstruiranje rešetke koja će optimalno, prema kriteriju krutosti, podnijeti zadano opterećenje, odnosno povezivanje određenih čvorova štapovima prikladnih dimenzija (čija ukupna masa nije veća od zadane) na takav način da je podataljivost konačne konstrukcije što je manja moguća, odnosno minimalna. Podatljivost je vrijednost recipročna krutosti i govori nam kolika će se deformacija javiti pri opterećenju jediničnom silom.



Slika 1.2. Stvarni primjeri rešetkastih konstrukcija [1].

Optimiranje konstrukcija prema fizikalnom značenju projektnih varijabli dijeli se na: optimiranje dimenzija, optimiranje oblika i optimiranje topologije. Postoje dva tipa topološkog optimiranja, a to su optimiranje diskretnih struktura i kontinuirano topološko optimiranje. Rešetkaste konstrukcije spadaju u diskretne konstrukcije. Za diskretne konstrukcije problem optimalne topologije sastoji se u određivanju optimalnog broja, pozicija i međusobne povezanosti strukturnih elemenata. Proces optimiziranja započinje sa inicijalnom strukturom (engl. *ground structure*) koja je okarakterizirana velikim brojem elemenata (štapova u rešetci) i velikim stupnjem povezanosti čvorova, što znači da je svaki čvor povezan sa velikim brojem čvorova kao što je prikazano na slici 1.3. Inicijalna mreža se tijekom procesa modificira na način da se eliminiraju elementi koji nisu optimalno iskorišteni. [2]



Slika 1.3. Inicijalna struktura (tzv. ground strucure)

2. Teorijske osnove

U ovom poglavlju biti će objašnjeno što je to linearno, a što semidefinitno programiranje, te će biti dani opći oblici njihovih zapisa.

2.1. Linearno programiranje

Linearno programiranje je poseban slučaj matematičkog programiranja. Ova metoda programiranja promatra probleme u kojima se linearna funkcija cilja mora optimizirati (maksimizirati ili minimizirati) uz uvjete ili ograničenja dana u obliku linearnih jednadžbi i/ili nejednadžbi. Linearno programiranje je metoda optimizacijskog programa sljedeće forme:

$$min\{c^T x | Ax \geq b\} \tag{1}$$

Gdje je:

- $x \in \mathbb{R}^n$ projektna varijabla,
- $c \in \mathbb{R}^n$ zadani vektor koeficijenata funkcije cilja,
- A zadana matrica krutosti (dimenzija m x n),
- $b \in \mathbb{R}^m$ slobodni član ograničenja i
- $\mathbb{R}$ je skup realnih brojeva.[3]

2.2. Semidefinitno programiranje

Semidefinitno programiranje bavi se optimizacijom linearne funkcije uz ograničenje da je afina kombinacija simetričnih matrica pozitivno semidefinitna. Takvo ograničenje je konveksno ograničenje te zbog toga semidefinitno programiranje spada u područje konveksnog programiranja.

Opći oblik semidefinitnog programa:

$$\min c^T x \tag{2}$$

Uz ograničenje: $F(x) \prec 0$ $(F(x) \succ 0, F(x) \preccurlyeq 0, F(x) \succcurlyeq 0)$

$$Ax = b.$$

Gdje je:

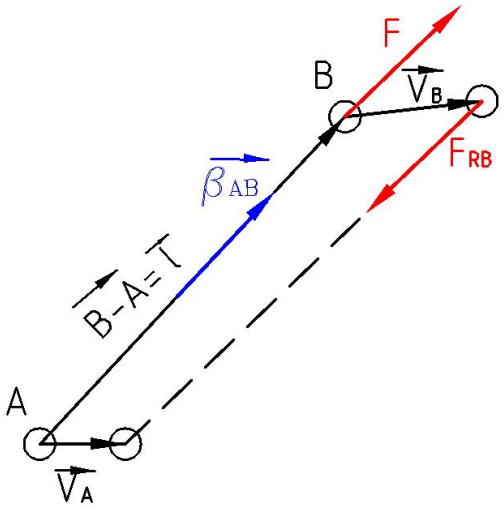
- $F(x) \prec 0$ – negativno definitna matrica,
- $F(x) \succ 0$ – pozitivno definitna matrica,
- $F(x) \preccurlyeq 0$- negativno semidefinitna matrica i
- $F(x) \succcurlyeq 0$ – pozitivno semidefinitna matrica.

$F(x)$ je simetrična matrica m x n koja ovisi o projektnim varijablama x. Varijable x moraju „linearno ulaziti „ (afino) u matricu F (x).

3. Formulacija problema (nominalni slučaj)

U ovom poglavlju biti će opisana matematička formulacija TTD (engl. *Truss Topology Design*) problema.

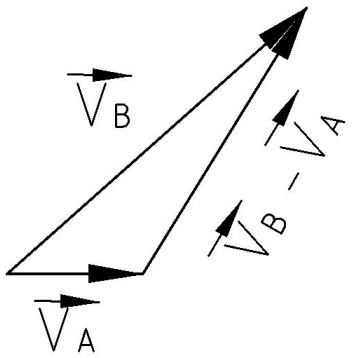
Najprije je potrebno razmotriti što se događa u štapu koji je opterećen. Na slici 3.1. prikazan je štap AB u neopterećenom položaju i njegova deformacija koja je nastala zbog primijenjenog opterećenja (crtkana linija).



Slika 3.1. Štap u neopterećenom i opterećenom stanju [3]

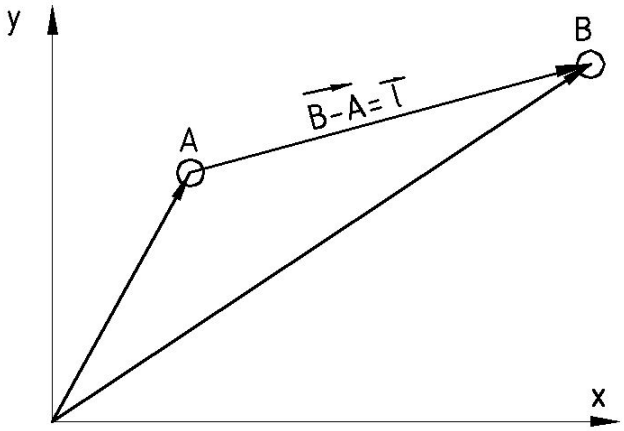
Prema [3] produljenje štapa dl zbog opterećenja je projekcija $(\overrightarrow{v_B - v_A})$ u smjeru štapa:

$$dl = \frac{(\overrightarrow{v_B - v_A})^T \cdot (\overrightarrow{B-A})}{||B-A||}, \text{odnosno } dl = \frac{(\overrightarrow{v_B - v_A})^T \cdot \vec{l}}{l}. \quad (3)$$



Slika 3.2. Vektori pomaka čvora

Gdje je $\frac{(\overrightarrow{B-A})}{\|\overrightarrow{B-A}\|}$ jedinični vektor štapa AB, v_B je pomak čvora B, a v_A pomak čvora A.



Slika 3.3. Štap AB u kordinatnom sustavu

Sila u štapu zbog produljenja prema Hookeovom zakonu:

$$\sigma = \varepsilon \cdot E \quad \left[\frac{N}{mm^2}\right]. \tag{4}$$

$$\sigma = \frac{F}{A} \qquad \left[\frac{N}{mm^2}\right]. \tag{5}$$

$$\varepsilon = \frac{dl}{l} \tag{6}$$

Gdje je:

- dl - apsolutno produljenje štapa,
- l –početna duljina štapa, odnosno: $\|\overrightarrow{B-A}\|$,
- $\vec{l}$ - vektor štapa AB, odnosno: $(\overrightarrow{B-A})$,
- E – Youngov modul elastičnosti,
- σ – naprezanje u štapu,
- ε – relativno produljenje štapa,
- A_{AB} – poprečni presjek štapa AB i
- t_{AB} – volumen štapa: $t_{AB} = A_{AB} \cdot l$.

Kombinacijom jednadžbi (4), (5) i (6) može se izraziti sila u štapu:

$$F = \frac{dl}{l} \cdot E \cdot A_{AB}. \tag{7}$$

Nadalje, uvrštavanjem izraza (3)u izraz (7) dobije se konačna jednadžba za silu u štapu:

$$F = \frac{\frac{(\overrightarrow{v_B - v_A})^T \cdot (\vec{l})}{l}}{l} \cdot E \cdot \frac{t_{AB}}{l}$$

$$F = \frac{E \cdot t_{AB} \cdot (\overrightarrow{v_B - v_A})^T \cdot \vec{l}}{l^3}. \tag{8}$$

Sila reakcije u točki B uzrokovana silom u štapu:

$$\frac{-F \cdot \vec{l}}{l} = \frac{E \cdot t_{AB} \cdot [(\overrightarrow{v_B - v_A})^T \cdot \vec{l}] \cdot \vec{l}}{l^4}$$

$$F_{RB} = -t_{AB} \cdot [(\overrightarrow{v_B - v_A})^T \cdot \beta_{AB}] \cdot \beta_{AB} \tag{9}$$

$$\beta_{AB} = \frac{\sqrt{E} \cdot \vec{l}}{l^2} \tag{10}$$

Vektor β_{AB} je vektor u smjeru od čvora A prema čvoru B veličine $\frac{\sqrt{E}}{l}$. On ovisi o poziciji čvorova povezanih štapovima, a neovisan je o opterećenju i o konstrukciji.

Potencijalna energija pohranjena u štapu posljedica je postojanja produljenja. Ona iznosi pola od umnoška sile u štapu i produljenja štapa, pa prema tome vrijedi:

$$\frac{F \cdot dl}{2} = \frac{E \cdot t_{AB} \cdot (\overrightarrow{v_B - v_A})^T \cdot \vec{l} \cdot (\overrightarrow{v_B - v_A})^T \cdot \vec{l}}{2 \times ||B - A||^4}$$

$$E_p = \frac{1}{2} t_{AB} \cdot [(\overrightarrow{v_B - v_A})^T \cdot \beta_{AB}]^2 \quad (11)$$

Prostor R^m virtualnih pomaka štapova definiran je kao direktna suma pomaka slobodnih čvorova. Slobodni čvorovi mogu imati pomak u x ili y smjeru, a čvrsti čvorovi (čvrsti oslonci) nemaju pomak ni u smjeru osi x ni u smjeru osi y. M_f označava broj slobodnih čvorova, a m je prema tome ili $2M_f$ ili $3M_f$ ovisno o tome da li se razmatra ravninski ili prostorni slučaj rešetke. Vektor v predstavlja pomak mreže čvorova i element je R^m. On sadrži pomake slobodnih čvorova, odnosno njihovu x i y komponentu, ako je ravninski slučaj ili njihovu x, y i z komponentu ako je prostorni slučaj.

Odnosno:

$$v[\nu] = \begin{bmatrix} v_{\nu,x} \\ v_{\nu,y} \end{bmatrix}. \quad (12)$$

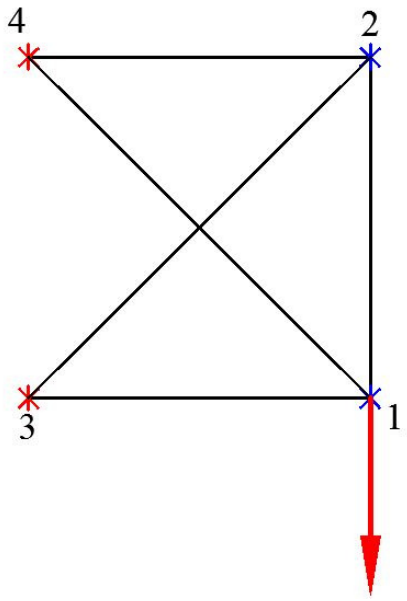
Gdje ν označava broj čvora.

Vektor b_i , koji će se kasnije u radu koristiti za formulaciju matrice krutosti, također je element R^m. Definiran je na sljedeći način:

- *i* označava broj štapa,
- ν označava broj čvora,
- ako je ν slobodan i označava drugi čvor štapa b_i [ν]=$\beta_{AiBi,}$
- ako je ν slobodan i označava prvi čvor štapa b_i [ν]= - β_{AiBi} i
- u svim ostalim slučajevima b_i[ν]=0.

Primjer 1:

Promatramo slučaj sa slike 3.4. Slobodni čvorovi označeni su plavom bojom (1 i 2), a čvrsti čvorovi crvenom bojom (3 i 4).



Slika 3.4. Primjer rešetkaste konstrukcije

Vektor pomaka v za slučaj sa slike 3.4., odnosno za dva slobodna čvora i ravninsku rešetku definiran je kao:

$$v = \begin{bmatrix} v_{1,x} \\ v_{1,y} \\ v_{2,x} \\ v_{2,y} \end{bmatrix}. \tag{13}$$

Odnosno $v[1] = \begin{bmatrix} v_{1,x} \\ v_{1,y} \end{bmatrix}$, $a\ v[2] = \begin{bmatrix} v_{2,x} \\ v_{2,y} \end{bmatrix}$.

Gdje $v_{1,x}$ označava pomak čvora 1 u smjeru osi x, $v_{1,y}$ pomak čvora 1 u smjeru osi y, $v_{2,x}$ označava pomak čvora 2 u smjeru osi x i $v_{2,y}$ pomak čvora 2 u smjeru osi y.

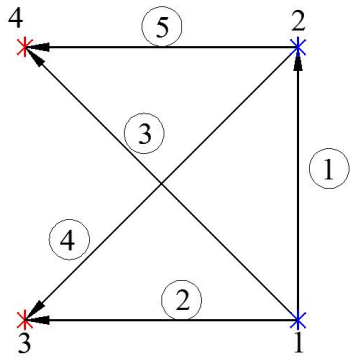
Analogno tome definiran je i vektor sile F, pa je za isti slučaj F:

$$F = \begin{bmatrix} F_{1,x} \\ F_{1,y} \\ F_{2,x} \\ F_{2,y} \end{bmatrix}. \tag{14}$$

Stvarni vektor sile za primjer na slici 3.4. glasi:

$$F = \begin{bmatrix} 0 \\ 2000 \\ 0 \\ 0 \end{bmatrix}.$$

Za primjer sa slike 3.4. uzet ćemo štap koji povezuje čvorove 1 i 2 i pretpostavit ćemo da njegov vektor gleda prema čvoru 2 što je važno za određivanje koji čvor je prvi čvor štapa, a koji drugi. Vektor *b* za štap 1 u čvoru 1 glasi*:* $b_1[1] = \begin{bmatrix} -\beta_{12x} \\ -\beta_{12y} \end{bmatrix}$, a vektor b za štap 1 u čvoru 2*:* $b_1[2] = \begin{bmatrix} \beta_{12x} \\ \beta_{12y} \end{bmatrix}$. Vektor *b* za štap 1 glasio bi $b_1 = \begin{bmatrix} -\beta_{12x} \\ -\beta_{12x} \\ \beta_{12x} \\ \beta_{12y} \end{bmatrix}$.



Slika 3.5. Označavanje štapova

Tako konstruirani vektori za sve štapove iz primjera prikazani su u tablici 1.

Tablica 1: Konstruiranje vektora b za sve postojeće štapove

	Vektor b u početnom čvoru štapa	*Vektor b u završnom čvoru štapa*	*Vektor b*
ŠTAP 1 *-povezuje 1 i 2 čvor*	$\begin{bmatrix}-\beta_{12x}\\-\beta_{12y}\end{bmatrix}$.	$\begin{bmatrix}\beta_{12x}\\\beta_{12y}\end{bmatrix}$.	$\begin{bmatrix}-\beta_{12x}\\-\beta_{12y}\\\beta_{12x}\\\beta_{12y}\end{bmatrix}$.
ŠTAP 2 *-povezuje 1 i 3 čvor*	$\begin{bmatrix}-\beta_{13x}\\-\beta_{13y}\end{bmatrix}$.	$\begin{bmatrix}0\\0\end{bmatrix}$.	$\begin{bmatrix}-\beta_{13x}\\-\beta_{13y}\\0\\0\end{bmatrix}$.
ŠTAP 3 *-povezuje 1 i 4 čvor*	$\begin{bmatrix}-\beta_{14x}\\-\beta_{14y}\end{bmatrix}$.	$\begin{bmatrix}0\\0\end{bmatrix}$.	$\begin{bmatrix}-\beta_{14x}\\-\beta_{14y}\\0\\0\end{bmatrix}$.
ŠTAP 4 *-povezuje 2 i 3 čvor*	$\begin{bmatrix}-\beta_{23x}\\-\beta_{23y}\end{bmatrix}$.	$\begin{bmatrix}0\\0\end{bmatrix}$.	$\begin{bmatrix}-\beta_{23x}\\-\beta_{23y}\\0\\0\end{bmatrix}$.
ŠTAP 5 *-povezuje 2 i 4 čvor*	$\begin{bmatrix}-\beta_{24x}\\-\beta_{24y}\end{bmatrix}$.	$\begin{bmatrix}0\\0\end{bmatrix}$.	$\begin{bmatrix}-\beta_{24x}\\-\beta_{24y}\\0\\0\end{bmatrix}$.

Formulacija izraza (9) za čvor 2 (reakcija u štapu 1 u čvoru 2):

$F_{R2} = -V_1 \cdot [\begin{pmatrix}v_{2,x} - v_{1,x}\\v_{2,y} - v_{1,y}\end{pmatrix}^T \cdot \beta_{12}] \cdot \beta_{12}$. Gdje je vektor β_{12} vektor u smjeru štapa 1 veličine $\frac{\sqrt{E}}{l_1}$.

Sile reakcije i matrica krutosti

Iz (9), (10) i gore navedenih kriterija za definiranje vektora b_i proizlazi sljedeće: za svaki slobodan čvor ν komponenta sile reakcije uzrokovana pomakom u *i*- tom štapu iznosi:

$$-t_i(b_i^T \cdot v) \cdot b_i[v]. \tag{15}$$

Posljedično, ukupna reakcija sile u čvoru ν je:

$$-\sum_{i=1}^{n} t_i(b_i^T \cdot v) \cdot b_i[v] \quad . \tag{16}$$

Zbroj svih sila reakcije u čvorovima iznosi:

$$-\sum_{i=1}^{n} t_i(b_i^T \cdot v) \cdot b_i \quad = -\left[\sum_{i=1}^{n} t_i \, b_i b_i^T\right] v. \tag{17}$$

$$A(t) \cdot v = -f_r \tag{18}$$

A(t) predstavlja matricu krutosti rešetke. Ona je simetrična matrica dimenzije m x m koja linearno ovisi o volumenu štapova.

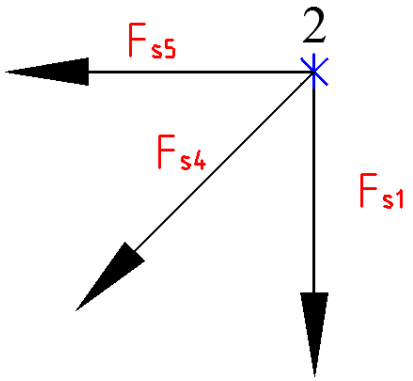
$$\sum_{i=1}^{n} t_i \, b_i b_i^T = A(t) \tag{19}$$

Primjer 1: nastavak

Formulacija izraza (15) za štap 1 u čvoru 2:

$$-t_1 \left(\begin{bmatrix} -\beta_{12x} \\ -\beta_{12y} \\ \beta_{12x} \\ \beta_{12y} \end{bmatrix}^T \cdot \begin{bmatrix} v_{1,x} \\ v_{1,y} \\ v_{2,x} \\ v_{2,y} \end{bmatrix} \right) \cdot \begin{bmatrix} \beta_{12x} \\ \beta_{12y} \end{bmatrix}.$$

Formulacija izraza (16) za štap 1 u čvoru 2:



Slika 3.6. Zbroj sila u čvoru 2

$$-\left[\left(t_1\left(\begin{bmatrix}-\beta_{12x}\\-\beta_{12y}\\\beta_{12x}\\\beta_{12y}\end{bmatrix}^T\cdot\begin{bmatrix}v_{1,x}\\v_{1,y}\\v_{2,x}\\v_{2,y}\end{bmatrix}\right)\cdot\begin{bmatrix}\beta_{12x}\\\beta_{12y}\end{bmatrix}\right)+\left(t_4\left(\begin{bmatrix}-\beta_{23x}\\-\beta_{23y}\\0\\0\end{bmatrix}^T\cdot\begin{bmatrix}v_{1,x}\\v_{1,y}\\v_{2,x}\\v_{2,y}\end{bmatrix}\right)\cdot\begin{bmatrix}-\beta_{23x}\\-\beta_{23y}\end{bmatrix}\right)\right.$$
$$\left.+\left(t_5\left(\begin{bmatrix}-\beta_{24x}\\-\beta_{24y}\\0\\0\end{bmatrix}^T\cdot\begin{bmatrix}v_{1,x}\\v_{1,y}\\v_{2,x}\\v_{2,y}\end{bmatrix}\right)\cdot\begin{bmatrix}-\beta_{24x}\\-\beta_{24y}\end{bmatrix}\right)\right].$$

Kako bi rešetka bila u ravnoteži sile reakcije moraju kompenzirati vanjske sile, a to nam daje sistem linearnih jednadžbi za određivanje pomaka rešetke uslijed vanjskog opterećenja f.

$$A(t)\cdot v = f \tag{20}$$

Potrebno je napisati izraz za podatljivost - potencijalnu energiju pohranjenu u rešetki u ravnotežnom stanju. Prema (11) i prema uvjetima za definiranje vektora b_i ta energija iznosi:

$$\frac{1}{2}\sum_{i=1}^{n} t_i\left[(v[v''(i)]-v[v'(i)])^T\beta_{A_iB_i}\right]^2=\frac{1}{2}\sum_{i=1}^{n} t_i(v^T b_i)^2$$

$$=\frac{1}{2}v^T\left[\sum_{i=1}^{n} t_i\, b_i b_i^T\right]v$$

$$= \frac{1}{2} v^T A(t) \cdot v$$

$$= \frac{1}{2} f^T \cdot v$$

Podatljivost rešetke pod opterećenjem f je:

$$Compl_f(t) = \frac{1}{2} f^T \cdot v \tag{21}$$

Primjer 1:

Konstruiranje matrice krutosti za primjer 1.

$$\sum_{i=1}^{n} t_i \, b_i b_i^T = A(t)$$

$$A = \begin{bmatrix} t_1 \cdot \beta_{12x}{}^2 + t_2 \cdot \beta_{13x}{}^2 + t_3 \cdot \beta_{14x}{}^2 + t_4 \cdot \beta_{23x}{}^2 + t_4 \cdot \beta_{23x}{}^2 + t_5 \cdot \beta_{24x}{}^2 & t_1 \cdot \beta_{12x} \cdot \beta_{12y} + t_2 \cdot \beta_{13x} \cdot \beta_{13y} + t_3 \cdot \beta_{14x} \cdot \beta_{14y} + t_4 \cdot \beta_{23x} \cdot \beta_{23x} + t_4 \cdot \beta_{23x} \cdot \beta_{23y} + t_5 \cdot \beta_{24x} \cdot \beta_{24y} & t_1 \cdot (-\beta_{12x}{}^2) & t_1 \cdot (-\beta_{12x} \cdot \beta_{12y}) \\ t_1 \cdot \beta_{12x} \cdot \beta_{12y} + t_2 \cdot \beta_{13x} \cdot \beta_{13y} + t_3 \cdot \beta_{14x} \cdot \beta_{14y} + t_4 \cdot \beta_{23x} \cdot \beta_{23x} + t_4 \cdot \beta_{23x} \cdot \beta_{23y} + t_5 \cdot \beta_{24x} \cdot \beta_{24y} & t_1 \cdot \beta_{12y}{}^2 + t_2 \cdot \beta_{13y}{}^2 + t_3 \cdot \beta_{14y}{}^2 + t_4 \cdot \beta_{23y}{}^2 + t_4 \cdot \beta_{23y}{}^2 + t_5 \cdot \beta_{24y}{}^2 & t_1 \cdot (-\beta_{12x} \cdot \beta_{12y}) & t_1 \cdot (-\beta_{12y}{}^2) \\ t_1 \cdot (-\beta_{12x}{}^2) & t_1 \cdot (-\beta_{12x} \cdot \beta_{12y}) & t_1 \cdot (-\beta_{12x}{}^2) & t_1 \cdot (\beta_{12x} \cdot \beta_{12y}) \\ t_1 \cdot (-\beta_{12x} \cdot \beta_{12y}) & t_1 \cdot (-\beta_{12y}{}^2) & t_1 \cdot (\beta_{12x} \cdot \beta_{12y}) & t_1 \cdot \beta_{12y}{}^2 \end{bmatrix}$$

S obzirom na poznavanje koordinata svakog čvora i modula elastičnosti lako je izračunati vektor β za svaki štap, a samim time i matricu krutosti rešetke. Kada se jednom izračuna matrica krutosti mogu se odrediti i pomaci čvorova u x i y smjeru.

4. Topološko optimiranje za nominalni slučaj

U ovom poglavlju biti će opisana formulacija linearnog programiranja za optimiranje topologije rešetkastih konstrukcija. Nadalje, bit će opisani problemi koji su se javili tijekom konstruiranja osnovne strukture kao i njihova rješenja.

4.1. Formulacija problema za linearno programiranje

Za zadanu osnovnu strukturu (korisnik zadaje mrežu čvorova, Youngov modul elastičnosti, volumen rešetkaste konstrukcije, opterećenja) potrebno je pronaći rešetku $t=(t_i....t_n)$ sa nenegativnim volumenima (t_i je volumen i-tog štapa) koji zadovoljava ograničenje:

$$\sum_{i=1}^{n} t_i \leq w \tag{22}$$

i s minimalnom mogućom podatljivošću $Compl_f(t)$ s obzirom na opterećenje f. [3]

Gdje w označava ukupni volumen konstrukcije zadan od strane korisnika. Iako su ograničenja $t \geq 0, \sum_{i=1}^{n} t_i \leq w$ linearna, funkcija cilja $Compl_f(t)$ definirana izrazima (19), (20) i (21) je nelinearna. Problem topološke optimizacije rešetke može se reformulirati i u prikladnom obliku svesti na linearno programiranje. Prema (10) i navedenim kriterijima za vektor b_i naprezanje u štapu *i* je jednostavna funkcija vektora pomaka:

$$s_i = |b_i^T \cdot v|. \tag{23}$$

Za zadanu strukturu i opterećenje f potrebno je naći pomak v koji minimizira rad $f^T \cdot v$ pod djelovanjem opterećenja uz ograničenje da su sva naprezanja ≤ 1.

$$\min_v \{f^T \cdot v | \left| b_i^T \cdot v \right| \leq 1, i = 1, \dots \dots \dots n\} \tag{24}$$

Prema [3] dualni problem izrazu (24) je ekvivalentan problemu:

$$\min_{q_1 \dots q_n} \left\{ \sum_{i=1}^{n} |q_i| \mid \sum_{i=1}^{n} q_i b_i = f \right\} \tag{25}$$

Prema teoremu o dualnosti LP-a oba problema su riješiva sa zajedničkom optimalnom vrijednošću w_*. Neka je v^*optimalno rješenje izraza (24) i neka q^* bude optimalno

rješenje izraza (25) pod pretpostavkom da je $f \neq 0$ vrijedi da je :$w_* = \sum_{i=1}^{n} |q_i^*| > 0$, tako da je vektor

$$t^*: t_i^* = \frac{w}{w_*} |q_i^*|, \qquad i = 1, \ldots \ldots \ldots n, \tag{26}$$

dobro definiran, ako je rešetka moguća (ako je t^* je nenegativan i zadovoljava ograničenje). Vektor t^* je optimalno rješenje TTD problema, a $v^+ = \frac{w_*}{w} v^*$ je odgovarajući pomak.

Kako bi mogli iskoristiti izraze (23), (24) i (25) potrebno je zapisati u obliku linearnog programa. Pa se izraz (24) zapisuje kao:

$$\min f^T \cdot v$$

Uz ograničenje: $\begin{bmatrix} b_1^T \\ -b_1^T \\ \vdots \\ b_n^T \\ -b_n^T \end{bmatrix} \cdot v \leq \begin{bmatrix} 1 \\ 1 \\ \vdots \\ 1 \\ 1 \end{bmatrix}$, i=1....n

Izraz (25) potrebno je zapisati na ovaj način:

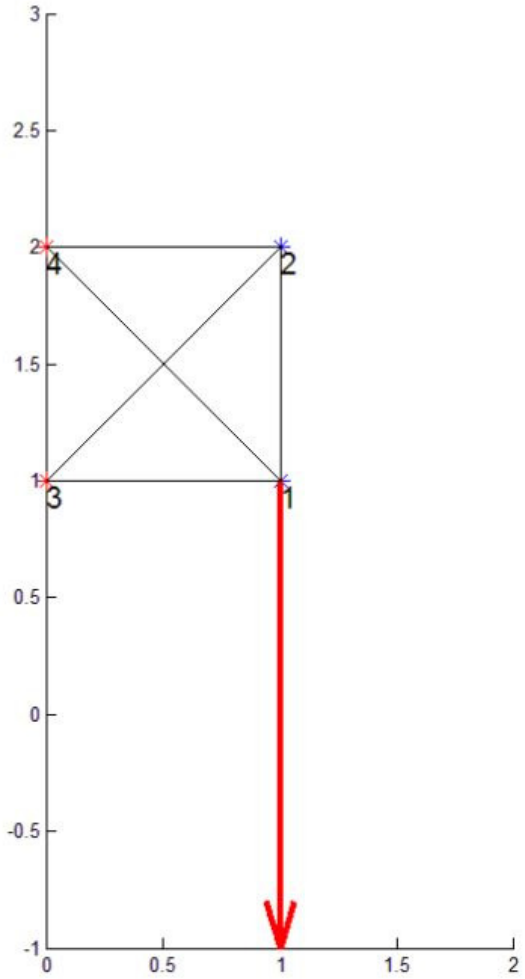
$$\min e^T \cdot \gamma$$

Uz ograničenje: $-\gamma \leq q \leq \gamma$, i $\sum_{i=0}^{n} q_i b_i = f$.

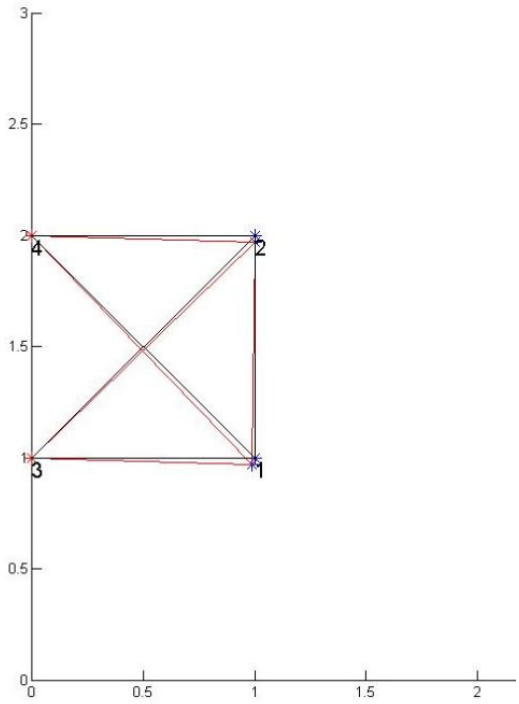
Gdje je: $e = \begin{bmatrix} 1 \\ \vdots \\ 1 \end{bmatrix}$, $q = \begin{bmatrix} q_1 \\ \vdots \\ q_n \end{bmatrix}$ i $\gamma = \begin{bmatrix} \gamma_1 \\ \vdots \\ \gamma_n \end{bmatrix}$.

Primjer 1

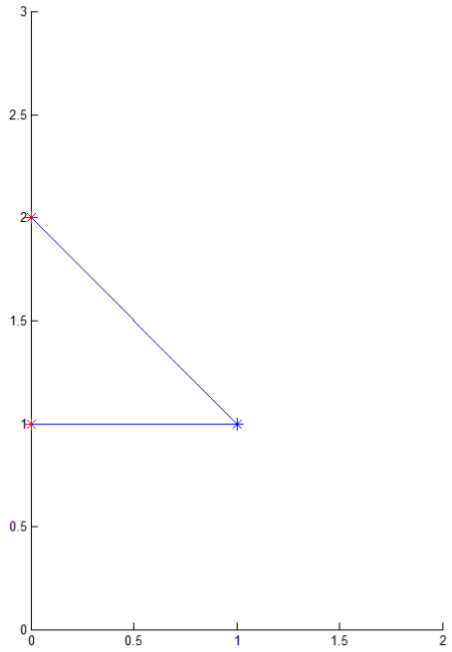
Zadana su dva slobodna čvora (1 i 2) i dva čvrsta čvora (3 i 4), te opterećenje u čvoru 1. Inicijalna mreža prikazana je na slici 4.1. Na slici 4.2. crvenom bojom prikazana je deformacija konstrukcije uzrokovana zadanom silom, a na slici 4.3. optimalna konstrukcija za ovakvu inicijalnu mrežu i zadano opterećenje.



Slika 4.1. Primjer1: Inicijalna mreža



Slika 4.2. Deformirana konstrukcija



Slika 4.3. Optimalna konstrukcija

4.2. Tehnički problemi i njihova rješenja

U ovom poglavlju biti će opisani načini kreiranja inicijalne strukture, problemi koji nastaju pri njenom konstruiranju i njihova rješenja.

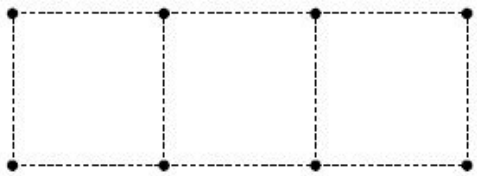
4.2.1. *Konstruiranje inicijalne strukture*

Prilikom generiranja inicijalne strukture svaka dva povezana čvora predstavljaju jedan štap. Kod ovakvog pristupa topološkom optimiranju inicijalna struktura, s obzirom na to da zapravo predstavlja potencijalne štapove, može imati veliki utjecaj na topologiju optimalne rešetke. Kako bi se uopće mogla generirati inicijalna struktura najprije je potrebno odrediti koji stupanj povezanosti čvorova će se koristiti. Razlikujemo tri stupnja povezanosti čvorova u inicijalnoj strukturi:

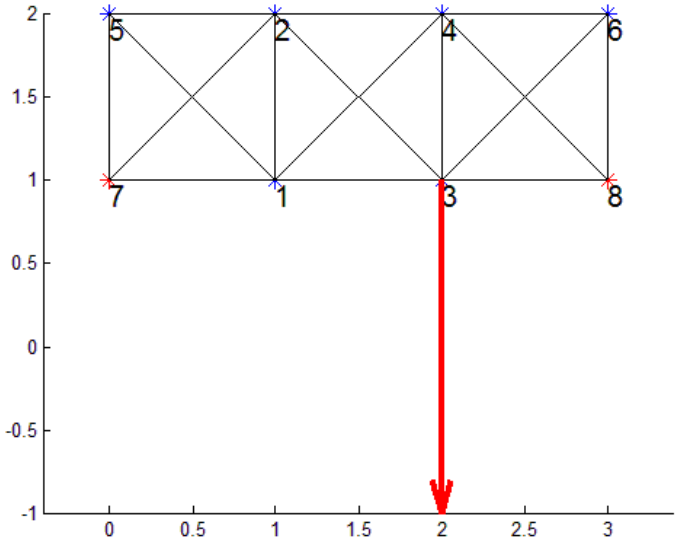
a) prvi stupanj povezanosti,
b) drugi stupanj povezanosti i
c) treći stupanj povezanosti, tzv. potpuna povezanost.

a) Prvi stupanj povezanosti čvorova

Kod topološkog optimiranja rešetkastih konstrukcija najčešće korištene inicijalne strukture su one s prvim i trećim stupnjem povezanosti [4]. Na slici 4.5. prikazana je inicijalna struktura s prvim stupnjem povezanosti, odnosno svaki od čvorova je povezan samo sa svojim susjedima (najbližim čvorovima). Ovakav stupanj povezanosti omogućava veću krutost nego ostali stupnjevi, a nedostatak mu je što odmah u početku eliminira štapove koji bi mogli biti dio optimalne rešetke.



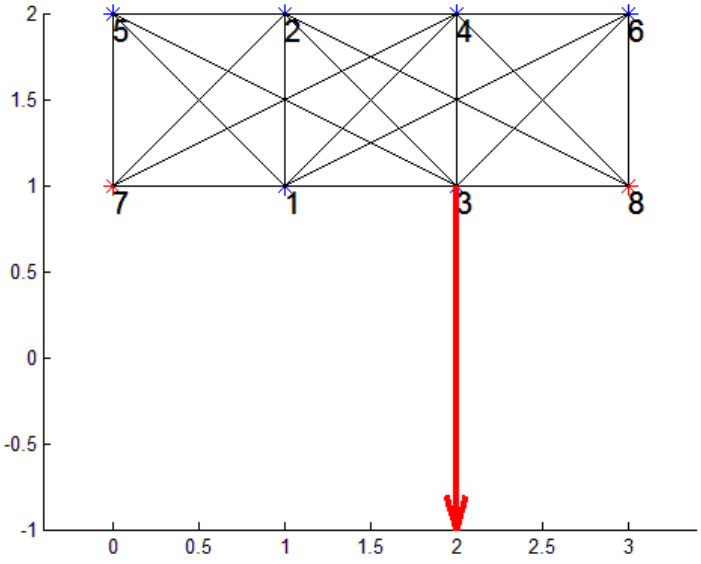
Slika 4.4. Osnovna mreža čvorova [5]



Slika 4.5. Prvi stupanj povezanosti čvorova

b) Drugi stupanj povezanosti čvorova

Na slici 4.6. prikazan je drugi stupanj povezanosti čvorova. Takav način stvaranja inicijalne mreže povezuje čvor sa njegovim susjednim čvorom, ali i sa susjedom njegovog susjeda.



Slika 4.6. Drugi stupanj povezanosti čvorova

c) Potpuna povezanost čvorova

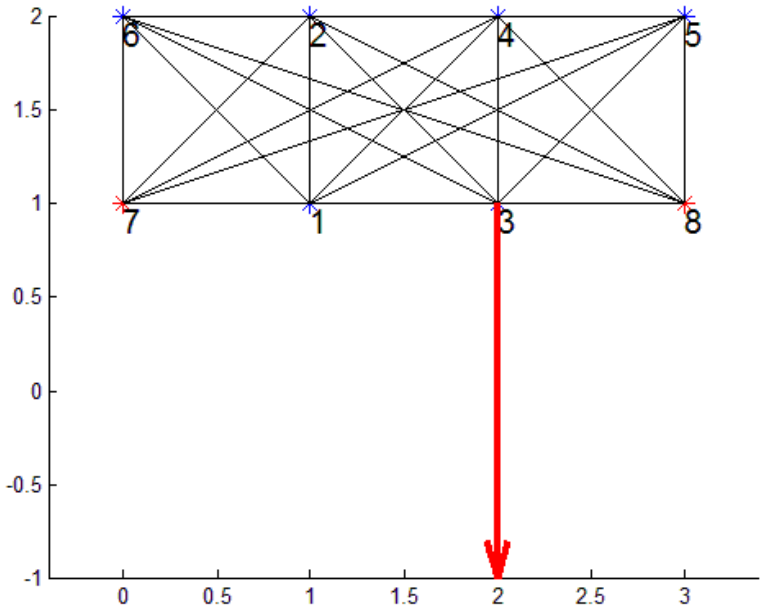
U slučaju potpune povezanosti čvorova, gdje je svaki čvor povezan sa svim ostalim čvorovima, broj potencijalnih štapova raste prema funkciji:

$$m = n(n-1).$$

Gdje je:

m- broj štapova i

n- broj čvorova.

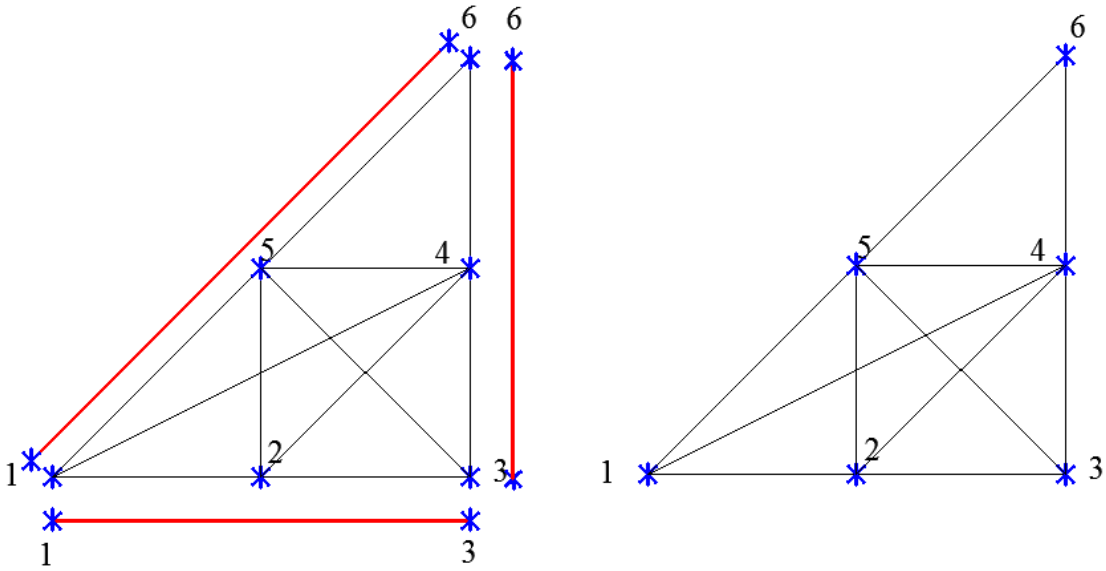


Slika 4.7. Potpuna povezanost čvorova

Inicijalna struktura, kao što je već rečeno, predstavlja potencijalne štapove te je potrebno riješiti sve moguće probleme kako bi se dobivena optimalna rešetka mogla izvesti i u stvarnosti. Problemi koji se javljaju kod konstruiranja inicijalne strukture su:

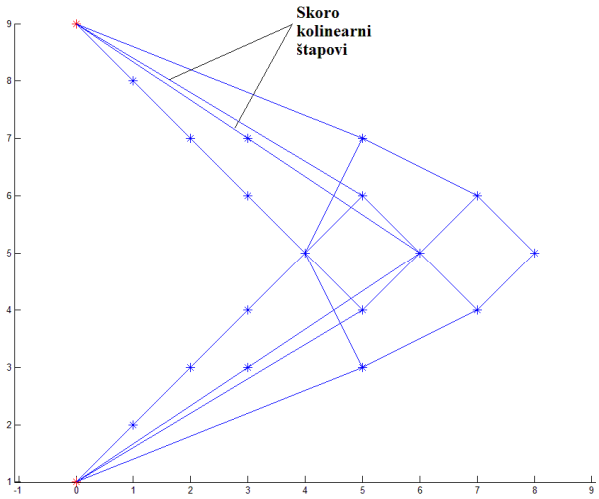
1. kolinearni štapovi, te
2. skoro kolinearni štapovi.

Tijekom povezivanja čvorova, a u svrhu konstruiranja inicijalne strukture nastaju kolinearni štapovi koje u stvarnosti nije moguće izvesti. Takvi štapovi prikazani su na slici 4.8., lijevo, crvenom bojom. Kako bi se riješio taj problem takvi štapovi eliminirani su prilikom stvaranja inicijalne strukture.



Slika 4.8. Inicijalna struktura sa kolinearnim štapovima (lijevo) i inicijalna struktura bez kolinearnih štapova (desno).

„Skoro kolinearni“ štapovi mogu nastati u svim stupnjevima povezanosti inicijalne mreže. Hoće li se „skoro kolinearni“ štapovi pojaviti ili ne ovisi o položaju čvorova, koji u ovom slučaju korisnik bira prema vlastitoj želji. Na slici 4.9. prikazana je već optimirana konstrukcija koja sadrži skoro kolinearne štapove. Problem nastaje kada bi se takva konstrukcija htjela izgraditi u stvarnosti, naime, unutar tako malog kuta bilo bi nemoguće izvesti dva štapa.



Slika 4.9. „Skoro kolinearni“ štapovi

4.2.2. *Eliminiranje kolinearnih i skoro kolinearnih štapova*

Kako bi optimizacija uopće imala u stvarnosti izvediv ishod najprije je potrebno provesti eliminaciju kolinearnih i skoro kolinearnih štapova. Na slici 4.10. prikazan je izvadak iz programskog koda napisanog u programskom paketu Matlab. Najprije se formiraju veze između svih čvorova, te se izračunava vektor smjera mogućeg štapa, a nakon toga izračunava se kut između vektora smjera mogućeg štapa i osi x. Programski kod zatim u varijablu „veze“ upisuje redom: prvi čvor štapa, drugi čvor štapa, duljinu mogućeg štapa i kut između vektora smjera štapa i osi x.

```
%% DEFINICIJA STAPOVA
% Generiranje stapova
bars=[];
stapovi=[];
% I. dio
% prolazak kroz sve slobodne cvorove
% i generiranje tablice svih mogucih veza
veze=[]; % [cvor1,cvor2,duljina,kut]
for i=1:N % kombinacija svakog slobodnog stapa
    for j=i+1:N+Nf % sa svima ostalima
        % vektor smjera moguceg stapa
        vek=[nodes(j,2),nodes(j,3)]-[nodes(i,2),nodes(i,3)];
        % kut izmedju tog vektora i x osi
        kut=atan2(det([vek;[3,0]]),dot(vek,[3,0]));
        % duljina vektora
        l=sqrt(vek(1)^2+vek(2)^2);
        veze(end+1,1:4)=[i,j,l,kut];
    end
end
```

Slika 4.10. Izvadak iz koda napisanog u programu Matlab

Varijabla nazvana „*SKS*“ predstavlja kriterij pomoću kojeg će se izbacivati kolinearni i skoro kolinearni štapovi prije optimizacije inicijalne strukture, a ona je zapravo kut između dva štapa koji dijele isti čvor i određuje ga korisnik. Varijabla "Lkut" je logički rezultat provjere da li je kut između dva štapa manji od „*SKS*“ kuta. Svi štapovi koji tome udovoljavaju (i koji imaju isti čvor 1 ili 2) brišu se iz popisa potencijalnih štapova. Kako je već navedeno korisnik sam određuje kut unutar kojeg ne želi da se nalaze dva štapa koja dijele isti čvor. Na slici 4.11. prikazan je izvadak iz programskog koda gdje je prikazano na koji način se iz inicijalne strukture eliminiraju kolinearani i „skoro

kolinearni“ štapovi. Izbacivanje skoro kolinearnih štapova prije optimizacije pri konstruiranju inicijalne mreže s većim brojem čvorova; znatno utječe na ubrzanje samog optimizacijskog procesa, a samim time doprinosi bržem izvršenju cijelog koda.

```
% II. dio
% sortiranje svih veza po najmanjoj duljini
% jer zelimo da najmanja duljina ima prednost
veze1=veze;
veze=sortrows(veze,[1,3]);
%RTveze2=veze;
% prolazak kroz sve prethodno generirane veze
% dodavanje u stapove i nakon dodavanja
% brisanje svih koje imaju isti kut unutar SKS podrucja
while size(veze,1)>0
    % dodaj stap
    bars(end+1,1:2)=veze(1,1:2);
    stapovi(end+1,1:4)=veze(1,1:4);
    veza=veze(1,1:4);
    % brisi sve moguce stapove s tim kutom i 1. cvorom
    Lkut=abs(veze(:,4)-veza(4))<SKSkut
    veze(Lkut & veze(:,1)==veza(1),:)=[]
    % brisi sve moguce stapove s tim kutom i 2. cvorom
    try % (potrebno kod zadnjeg prolaska)
        Lkut=abs(veze(:,4)-veza(4))<SKSkut
        veze(Lkut & veze(:,2)==veza(2),:)=[]
    end

end
```

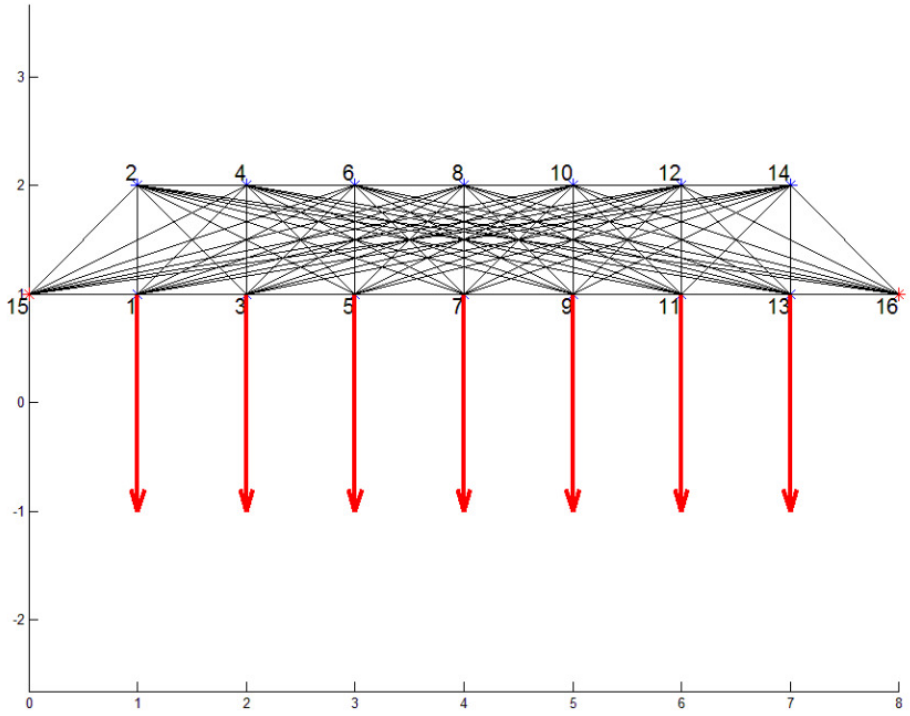
Slika 4.11. Izvadak iz koda napisanog u programu Matlab-rješavanje kolinearnosti

4.3. Motivacija za robusno optimiranje

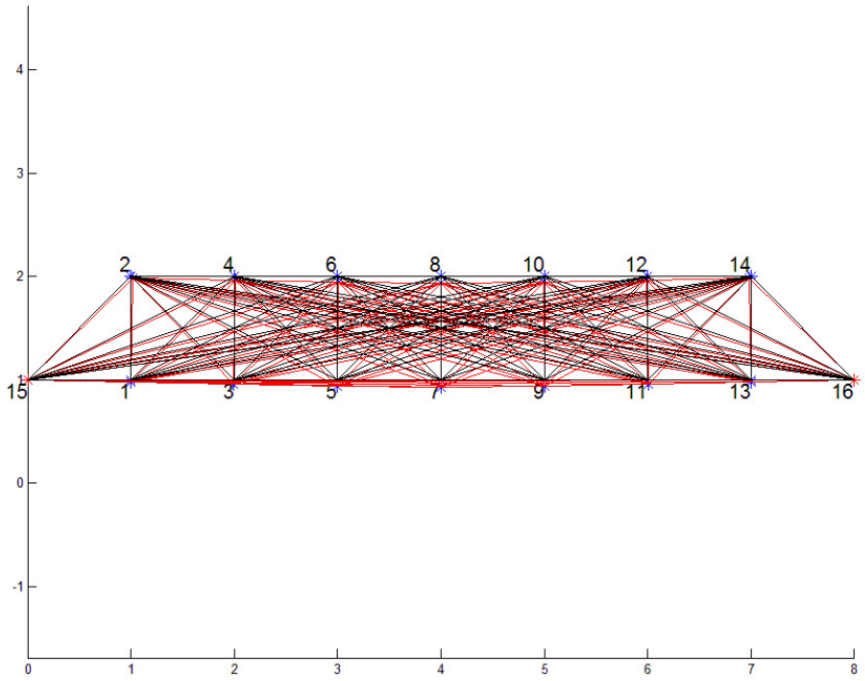
U ovom poglavlju bit će opisana potreba za robusnim optimiranjem rešetkastih konstrukcija.

Primjer 1

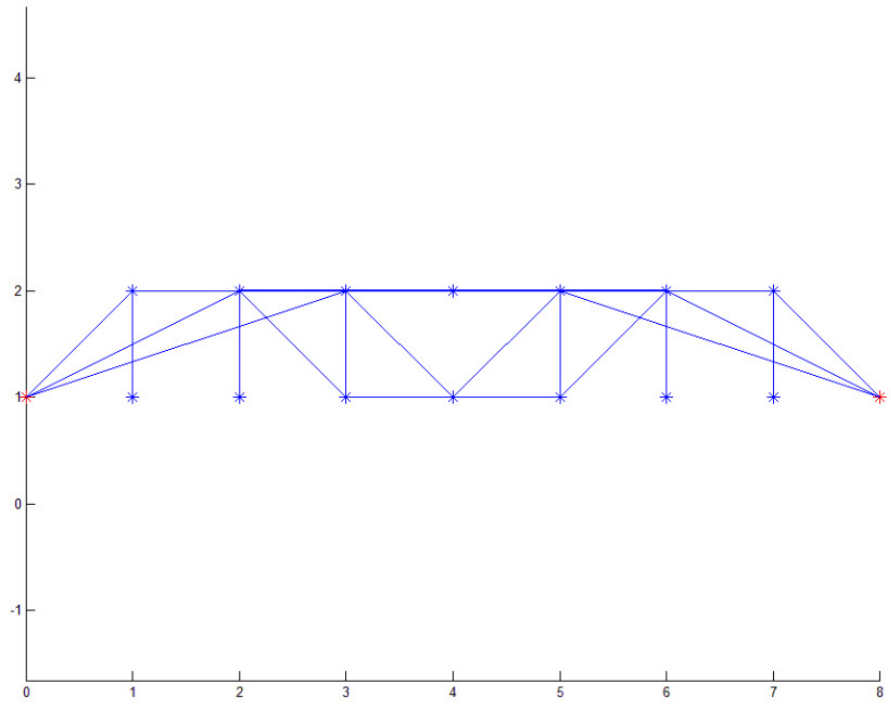
Zadana je mreža slobodnih čvorova 2*7 (2 - broj vertikalnih čvorova, 7 broj horizontalnih čvorova) i zadana su dva čvrsta čvora (15 i 16). Inicijalna mreža prikazana je na slici 4.12. Na slici 4.13. crvenom bojom prikazana je deformacija konstrukcije uzrokovana zadanim opterećenjem, a na slici 4.14. optimalna konstrukcija za ovakvu inicijalnu mrežu i zadano opterećenje. Podatljivost prikazane optimalne konstrukcije iznosi: 5,5.



Slika 4.12. Inicijalna mreža



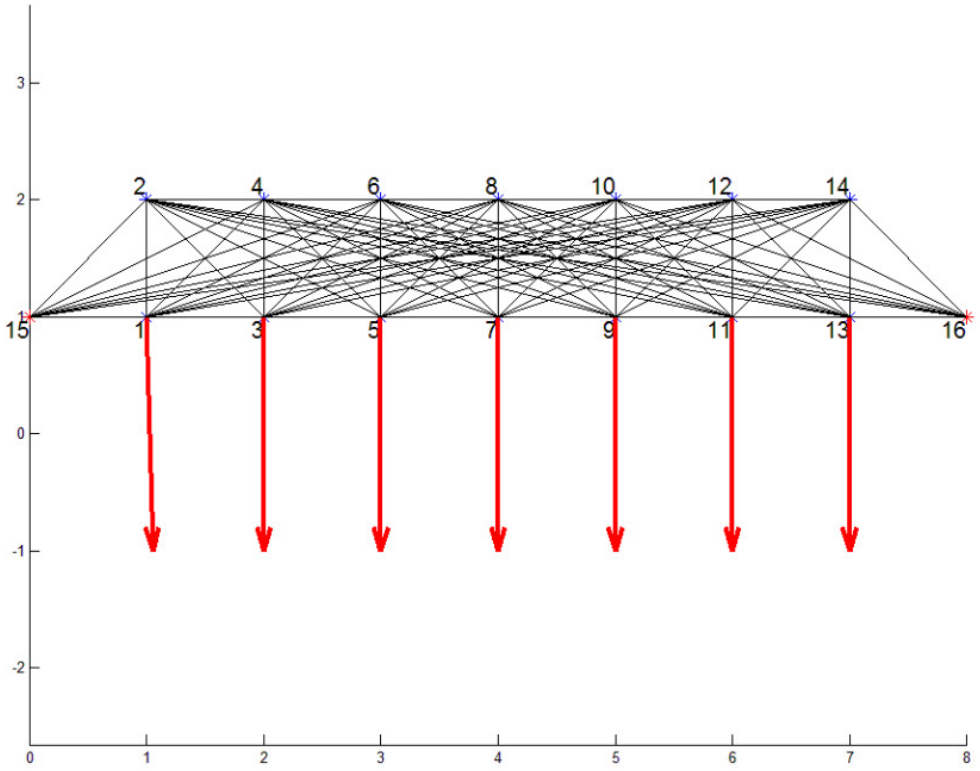
Slika 4.13. Deformirana mreža



Slika 4.14. Optimalna konstrukcija

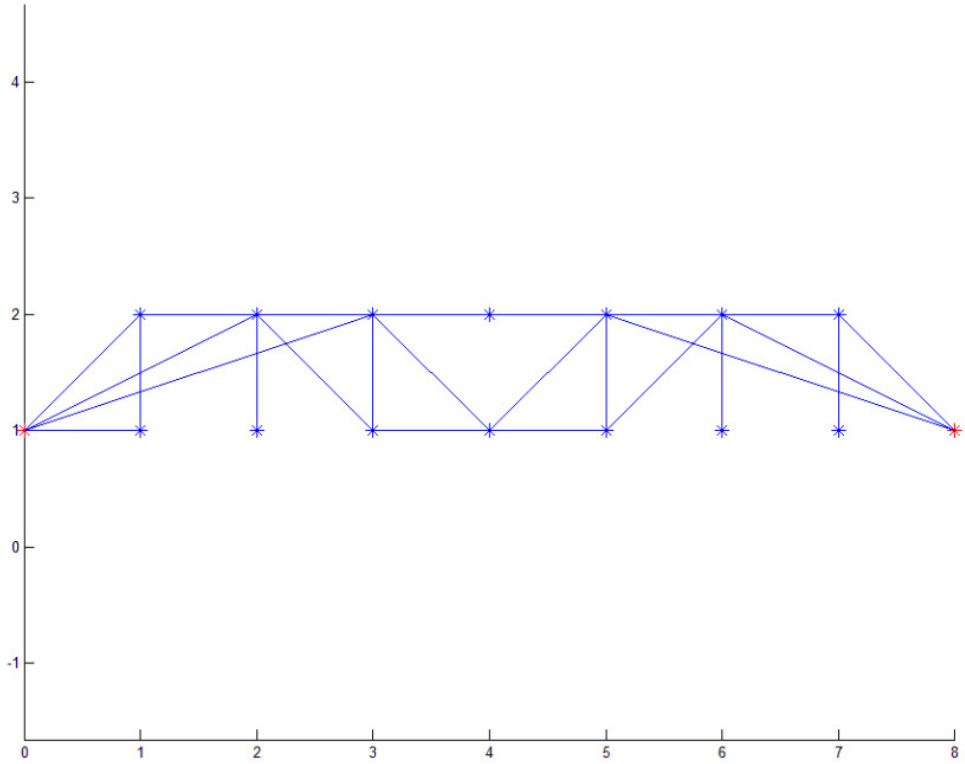
Kao što je vidljivo na slici 4.14. određeni štapovi ostaju „visjeti u zraku“ zbog toga što je u obzir uzeto samo nominalno opterećenje. Zbog postojanja takvih štapova

konstrukcija se smatra izrazito nestabilnom jer štap koji povezuje čvorove 2 i 1 može rotirati oko čvora 2 (isto vrijedi i za ostale takve štapove). Kada bi u čvoru 1 djelovala čak i mala nevertikalna sila podatljivost konstrukcije bi porasla i to bi zahtijevalo promjenu topologije konstrukcije. Kada bi sili u čvoru 1 u smjeru osi y (sve prikazane sile su u iznosu od 200 N) dodali silu u smjeru osi x od samo 5 N bila bi potrebna promjena topologije rešetke (podatljivost bi porasla na 3,851x 10^3), odnosno bio bi potreban dodatan štap koji bi povezivao čvorove 1 i 15. Dobro konstruirana rešetka trebala bi osigurati opravdanu krustost i prilikom nepredvidljivih malih opterećenja, a ne samo najbolju moguću krutost uslijed nominalnog opterećenja.



Slika 4.15. Inicijalna struktura za dodatno opterećenje

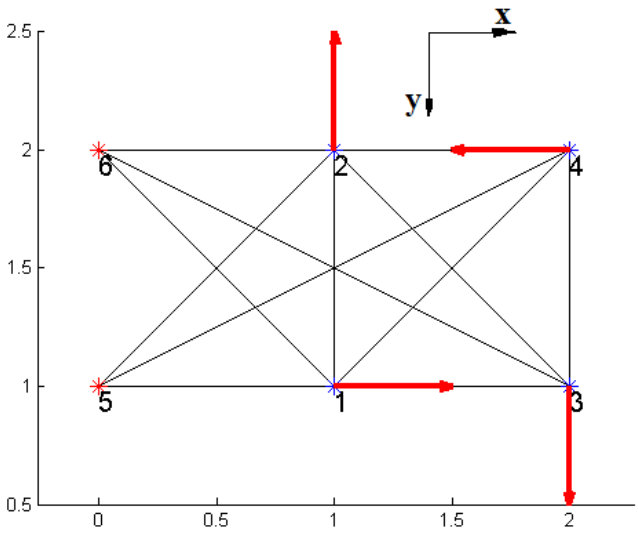
Na slici 4.15. prikazana je inicijalna struktura, a na slici 4.16. prikazana je optimalna konstrukcija za nametnuto dodatno opterećenje. Sa slike 4.16. vidiljiva je promjena topologije konstrukcije čiji je uzrok sila od samo 5 N u smjeru osi y.



Slika 4.16. Optimalna konstrukciija za dodatno opterećenje

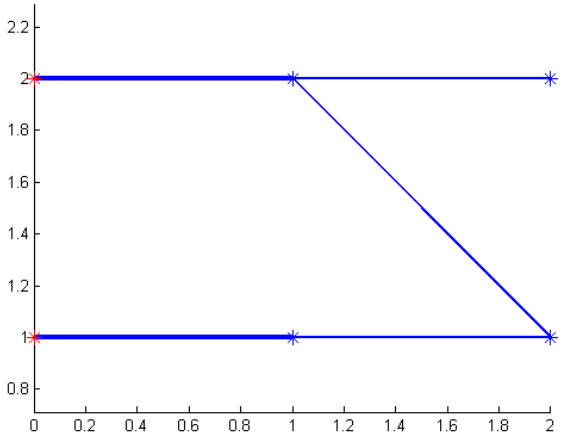
Primjer 2

Primjer je preuzet iz literature [6], te je „preveden“ u koordinatni sustav koji se koristi u ovom programskom kodu. Na slici 4.17. prikazana je inicijalna struktura sastavljena od četiri slobodna i 2 čvrsta čvora, te zadano opterećenje f. Optimalna podatljivost rešetke za ovaj slučaj iznosi 16,00. No, potrebno je primijetiti da je i u ovom primjeru optimalna rešetka potpuno nestabilna te da će i „mala“ ne horizotalna sila u čvoru 4 uzrokovati beskonačnu podatljivost. Kada bi sili u čvoru 4 dodali silu od samo 0.01 N u smjeru osi y došlo bi do beskonačnog porasta podatljivosti (1,517x10^5!).



Slika 4.17. Primjer 2: početna mreža [6]

Rešetkasta konstrukcija trebala bi biti konstruirana na način da ima „razumnu“ podatljivost čak i u slučaju nepredvidljivih malih opterećenja, a ne samo minimalnu moguću podatljivost u slučaju nominalnog opterećenja. Takav način optimiranja rešetkastih kostrukcija biti će opisan u sljedećem poglavlju.



Slika 4.18. Primjer 2: optimalna rešetka

5. Robusno topološko optimiranje

U ovom poglavlju biti će opisano kako nominalni problem topološkog optimiranja rešetkastih konstrukcija reformulirati u nominalni problem semidefinitnog optimiranja rešetkastih konstrukcija. Nadalje, bit će opisano što je to robusno optimiranje rešetkastih konstrukcija, te će biti prikazana formulacija problema za robusno optimiranje rešetkastih konstrukcija.

5.1. Semidefinitna reformulacija standardnog topološkog optimiranja

U ovom poglavlju cilj je formulirati nominalni problem topološkog optimiranja rešetkastih konstrukcija u obliku semidefinitnog programa.

Zadano je:

- $f \in \mathbb{R}^m$ – vanjsko opterećenje,
- $E_c(v) = \frac{1}{2} v^T A v$ - potencijalna energija pohranjena u konstrukciji kao rezultat pomaka *v*,
- $V=\mathbb{R}^m$ - linearni prostor virtualnih pomaka konstrukcije, i
- $\nu = \mathbb{R}^m$ – prostor kinematički dopustivih pomaka.

Statička ravnoteža konstrukcije uslijed vanjskog opterećenja *f* definirana je na sljedeći način: konstrukcija može nositi vanjsko opterećenje *f* ako i samo ako kvadratna forma

$$E_c{}^f(v) = \frac{1}{2} v^T A(t) v - f^T v \tag{27}$$

pomaka v ima svoj minimum u setu kinematički dopustivih pomaka, a ravnotežni pomaci minimiziraju $E_c{}^f(v)$. Negativna vrijednost minimuma vrijednosti $E_c{}^f(v)$ je takozvana podatljivost konstrukcije s obzirom na opterećenje f. Problem nominalnog topološkog optimiranja rešetkastih konstrukcija glasi:

Traži se *t* koji minimizira $Compl_f(t)$, gdje je:

$$Compl_f(t) = \sup_{v \in \nu} \left[f^T v - \frac{1}{2} v^T A(t) v \right]. \tag{28}$$

Matrica *A* je pozitivno definitna m x m matrica krutosti konstruirana kao:

$$A(t) = \sum_{i=1}^{n} t_i \, b_i b_i^T$$

Problem nominalnog topološkog optimiranja rešetkastih konstrukcija može se ekvivalentno zapisati kao :

$$\min \tau \tag{29}$$

Uz ograničenje:

$$Compl_f(t) \leq \tau \tag{30}$$

Ovaj je problem ekvivalentan je sljedećem problemu:

$$\min_{t,\tau} \tau \tag{31}$$

Uz ograničenje:

$$f^T v - \frac{1}{2} v^T A(t) v \leq \tau \qquad \text{za sve } v \in \mathcal{V}. \tag{32}$$

Kako bi se ovaj problem mogao riješiti pomoću semidefinitnog programiranja potrebno ga je zapisati u obliku semidefinitnog programa. Zapisivanje problema u obliku semidefinitnog programa prikazano je u sljedećih nekoliko koraka.

$$\frac{1}{2} v^T A(t) v - f^T v + \tau \geq 0 \text{ za sve } v \tag{33}$$

Izraz (33) mora vrijediti za sve moguće pomake v, te se on može zapisati pomoću izraza (34) i (35).

$$\begin{bmatrix} 1 \\ v \end{bmatrix}^T \begin{bmatrix} \tau & -\frac{1}{2} f^T \\ -\frac{1}{2} f & \frac{1}{2} A(t) \end{bmatrix} \begin{bmatrix} 1 \\ v \end{bmatrix} \geq 0 \text{ za sve } v \tag{34}$$

$$\begin{bmatrix} \tau & -\frac{1}{2} f^T \\ -\frac{1}{2} f & \frac{1}{2} A(t) \end{bmatrix} \succcurlyeq 0 \tag{35}$$

Sada se ovaj problem može zapisati u obliku semidefinitnog programa na sljedeći način:

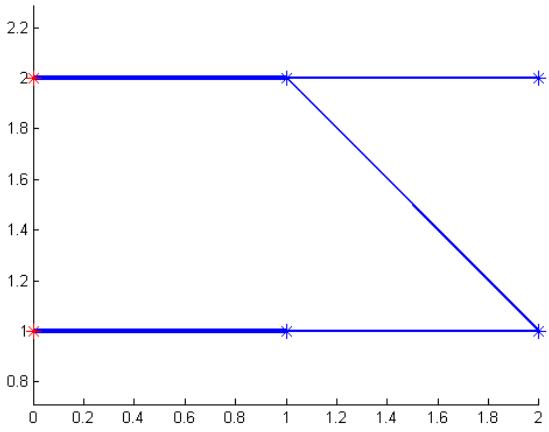
$$\min_{t,\tau} \tau \tag{36}$$

Uz ograničenje:

$$\begin{bmatrix} \tau & -\frac{1}{2} f^T \\ -\frac{1}{2} f & \frac{1}{2} A(t) \end{bmatrix} \succcurlyeq 0.$$

Kao primjer semidefinitnog programiranja promatra se primjer 2 iz poglavlja 4.3. Na slici 5.1. prikazana je rešetkasta konstrukcija dobivena semidefinitnom optimizacijom.

Usporedbom optimalnih konstrukcija dolazi se do zaključka da je optimalna struktura dobivena semidefinitnom optimizacijom jednaka optimalnoj strukturi dobivenoj linearnim optimiranjem na slici 4.18. Takav zaključak povlači pitanje zašto koristiti semidefinitno kada postoji linearno programiranje.



Slika 5.1. Semidefinitno programiranje – nastavak primjera 2

Odogovor na to pitanje nalazi se u potrebi za robusnim optimiranjem rešetkastih konstrukcija. Nominalni slučaj topološkog semidefinitnog optimiranja lako je proširiti u program za robusno semidefinitno topološko optimiranje. O tome će biti riječi u sljedećem poglavlju.

5.2. Robusna semidefinitna formulacija topološkog optimiranja

Cilj ovog poglavlja je formulirati semidefinitni program za robusno topološko optimiranje rešetkastih konstrukcija.

Prema [6] rešetka se smatra robusnom ako je opravdano kruta s obzirom na zadana, ali i mala nepredvidiva (u smjeru i veličini) opterećenja. Takva opterećenja mogu djelovati u bilo kojem aktivnom čvoru tj. u svim onim čvorovima koji su povezani s drugim čovorovima najmanje preko jednog štapa. Kako i njihov naziv govori takva opterećenja su nepredvidiva te je najveći izazov kako odrediti gdje (u kojim čvorovima) i s kojim intenzitetom će se takva opterećenja pojaviti.

Zadano je:

1) osnovna struktura m; n; $\{b_i \in \mathbb{R}^m\}_{i=1}^n$
2) dopušteni ukupni volumen w

   tj. $t = (t_1, \dots t_n) \in \mathbb{R}^n : \sum_{i=1}^n t_i \leq w,$

3) opterećenje: f ∈ $\mathbb{R}^m$, i
4) Ƒ ∈ $\mathbb{R}^m$ –sadrži nominalno opterećenje i nesigurnosti opterećenja svih čvorova.

Potrebno je naći konstrukciju koja je najkruća s obzirom na opterećenje Ƒ, tj. traže se volumeni t ∈ w koji maksimiziraju najgoru moguću podatljivost s obzirom na nominalno opterećenje f gdje je f ∈ Ƒ. Podatljivost konstrukcije s obzirom na nominalno opterećenje je definirana kao:

$$\min_{t\in w}\left\{Compl_{\mathrm{F}}(t)\equiv \sup_{f\in \mathrm{F}}\ \sup_{v\in \nu}\left[f^T v-\frac{1}{2}v^T A(t)v\right]\right\} \tag{37}$$

Problem je kako set opterećenja Ƒ modelirati tako da se ovaj slučaj može riješiti kao semidefinitni program. U slučaju robusnog semidefinitnog optimiranja konstrukcija s jednim setom nominalnog opterećenja set Ƒ je elipsoid, te vrijedi:

$$\mathrm{F}=\{\mathrm{f}=\mathrm{Qu}|\mathrm{u^T u}\le 1\},\mathrm{Q}\in \mathrm{M^{m,k}}. \tag{38}$$

Kako bi se problem robusnog topološkog optimiranja mogao riješiti pomoću semidefinitnog programiranja potrebno ga je zapisati u obliku semidefinitnog programa, što je prikazano u sljedećim koracima.

Iz izraza (37) proizlazi:

$$Compl_{\mathrm{F}}(t)\le \tau$$

$$f^T v-\tfrac{1}{2}v^T A(t)v\le \tau \text{ mora vrijediti za sve f} \in \mathrm{F} \text{ i za sve v} \in \nu \tag{39}$$

Izraz (39) ekvivalentan je sljedećem izrazu:

$$\frac{1}{2}v^T A(t)v-f^T v+\tau\ge 0 \tag{40}$$

Izraz (40) mora vrijediti za sve f ∈ Ƒ i za sve v ∈ ν pa se može zapisati kao:

$$\begin{bmatrix}1\\ v\end{bmatrix}^T\begin{bmatrix}\tau & -\frac{1}{2}f^T\\ -\frac{1}{2}f & \frac{1}{2}A(t)\end{bmatrix}\begin{bmatrix}1\\ v\end{bmatrix}\ge 0 \text{ mora vrijediti za sve f} \in \mathrm{F} \text{ i za sve v} \in \nu. \tag{41}$$

Ukoliko u prethodni izraz uvrstimo: $f=Qu$ izraz (41) zapisujemo u sljedećem obliku:

$$\begin{bmatrix}\tau & -\frac{1}{2}\mathrm{u^T Q^T}\\ -\frac{1}{2}Qu & \frac{1}{2}A(t)\end{bmatrix}\succcurlyeq 0 \qquad \text{vrijedi za sve } \mathrm{u^T u}\le 1$$

$$\begin{bmatrix} \tau & -\frac{1}{2}\mathrm{u}^{\mathrm{T}}\mathrm{Q}^{\mathrm{T}} \\ -\frac{1}{2}Qu & \frac{1}{2}A(t) \end{bmatrix} \succcurlyeq 0 \text{ vrijedi za sve } \mathrm{u}^{\mathrm{T}}\mathrm{u} = 1 \tag{42}$$

Izraz (42) ekvivalentan je sljedećem izrazu:

$$\begin{bmatrix} \tau & -\frac{1}{2}\frac{\mathrm{w}^{\mathrm{T}}}{\|\mathrm{w}\|_2}\mathrm{Q}^{\mathrm{T}} \\ -\frac{1}{2}Q\frac{\mathrm{w}^{\mathrm{T}}}{\|\mathrm{w}\|_2} & \frac{1}{2}A(t) \end{bmatrix} \geq 0 \quad \text{za sve w} \neq 0. \tag{43}$$

Gdje je $\frac{\mathrm{w}^{\mathrm{T}}}{\|\mathrm{w}\|_2}$ jedinični vektor u. Pomnožimo li (43) sa $\mathrm{w}^{\mathrm{T}}\mathrm{w}$ (gdje je $\mathrm{w}^{\mathrm{T}}\mathrm{w} = \|\mathrm{w}\|_2{}^2$) dobivaju se sljedeći izrazi:

$$\mathrm{w}^{\mathrm{T}}\mathrm{w}\begin{bmatrix} \tau & -\frac{1}{2}\frac{\mathrm{w}^{\mathrm{T}}}{\|\mathrm{w}\|_2}\mathrm{Q}^{\mathrm{T}} \\ -\frac{1}{2}Q\frac{\mathrm{w}^{\mathrm{T}}}{\|\mathrm{w}\|_2} & \frac{1}{2}A(t) \end{bmatrix} \geq 0 \quad \text{za sve w} \neq 0,$$

$$\begin{bmatrix} \tau\,\mathrm{w}^{\mathrm{T}}\mathrm{w} & -\frac{1}{2}\,\mathrm{w}^{\mathrm{T}}\mathrm{Q}^{\mathrm{T}}\sqrt{\mathrm{w}^{\mathrm{T}}\mathrm{w}} \\ -\frac{1}{2}Q\,\mathrm{w}^{\mathrm{T}}\sqrt{\mathrm{w}^{\mathrm{T}}\mathrm{w}} & \frac{1}{2}A(t)\sqrt{\mathrm{w}^{\mathrm{T}}\mathrm{w}} \end{bmatrix} \geq 0 \quad \text{za sve w} \neq 0 \tag{44}$$

Izraz (44) vrijedi za sve w ≠0 pa se može zapisati kao:

$$\begin{bmatrix} w \\ -\sqrt{\mathrm{w}^{\mathrm{T}}\mathrm{w}} \end{bmatrix}^{\mathrm{T}} \begin{bmatrix} \tau I & \frac{1}{2}\mathrm{Q}^{\mathrm{T}} \\ \frac{1}{2}Q & \frac{1}{2}A(t) \end{bmatrix} \begin{bmatrix} w \\ -\sqrt{\mathrm{w}^{\mathrm{T}}\mathrm{w}} \end{bmatrix} 0 \geq 0 \quad \text{za sve w} \neq 0 \tag{45}$$

Semidefinitni oblik programa za robusno optimiranje glasi:

$$\min_{t,\tau} \tau$$

Uz ograničenje:

$$\begin{bmatrix} 2\tau I & \mathrm{Q}^{\mathrm{T}} \\ Q & A(t) \end{bmatrix} \succcurlyeq 0$$

$$\sum \mathrm{t_i} \leq \mathrm{w}$$

$$\mathrm{t_i} \geq 0 \quad \text{za sve i.}$$

Gdje je I jedinična matrica, a Q je set nominalnog opterećenja i nepredvidivih malih opterećenja. Sada još samo ostaje definirati način određivanja matrice Q. Prema [6]

najbolji način za određivanje nepredvidivih sila u čvorovima je da taj da se u ishodište postavi elipsoid:

$$M = QW_q \equiv \{Qe | e \in R^q, \mathrm{e^T e} \leq 1\}$$

Matrica Q je veličine n x q , a Wq je jedinična euklidska kugla u Rq. U slučaju robusnosti nominalno opterećenje element je matrice M iz čega slijedi da ukoliko f □ M ima silu koja djeluje u nekom čvoru taj čvor će sigurno biti prisutan u optimalnoj konstrukciji. Nema smisla tražiti najveću krutost konstrukcije koja bi nosila „mala" nepredvidiva opterećenja u svim čvorovima inicijalne strukture jer bi tada svi čvorovi bili prisutni u optimalnoj rešetkastoj konstrukciji. Zbog toga je potrebno vrlo oprezno odabrati matricu opterećenja Q. Najprije je potrebno ustanoviti koji čvorovi će sigurno biti prisutni u optimalnoj rešetci,a takvi čvorovi odredit će se na sljedeći način:

1. U prvom koraku postavit će se semidefinitni program za nominalno opterećenje – u obzir se uzima samo nominalno opterećenje, te će nakon takve optimizacije biti poznati čvorovi koji se nalaze u optimalnoj konstrukciji,
2. U drugom koraku optimalni čvorovi iz prvog koraka sada čine novu inicijalnu strukturu, a nova inicijalna struktura optimizira se s obzirom na nominalna opterećenja i na "mala" nepredvidiva opterećenja za koje se smatra da djeluju u svim optimalnim čvorovima iz prvog koraka.

Prema [6] elipsoid M trebao bi sadržavati:

- set zadanih opterećenja f i
- kuglu $B = \{f \in R^q | f^T f \leq r^2\}$ koja predstavlja sva "nepredvidiva" opterećenja unaprijed određenog polumjera r.

Nadalje, LEMMA 2.1 [6] kaže: Pod određenim pretpostavkama elipsoidni razvoj opterećenja f i B je:

$$M = QW_q, \qquad Q = \left[f; re_1; re_2; re_3; re_{q-k};\right]$$

Gdje je $e_1; e_2; e_3; e_{q-k}$ ortonormalna baza ortogonalnom komplementu L(f) u R^q.

U ovom radu radijus kugle definiran je kao:

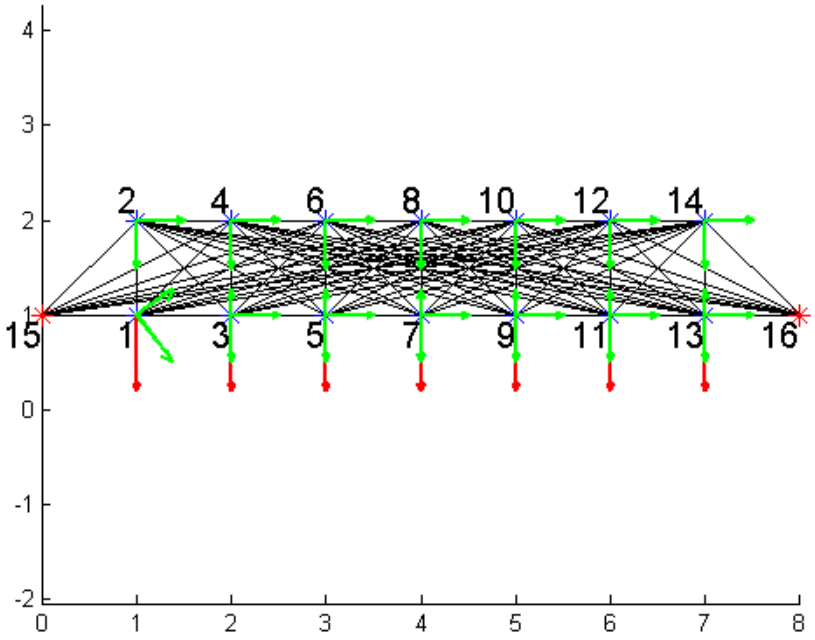
$$r = \sqrt{0.1 f^T f}$$

5.3. Primjeri robusnog semidefinitnog topološkog programiranja

U ovom poglavlju bit će prikazano nekoliko primjera robusnog topološkog optimiranja.

Primjer 1:

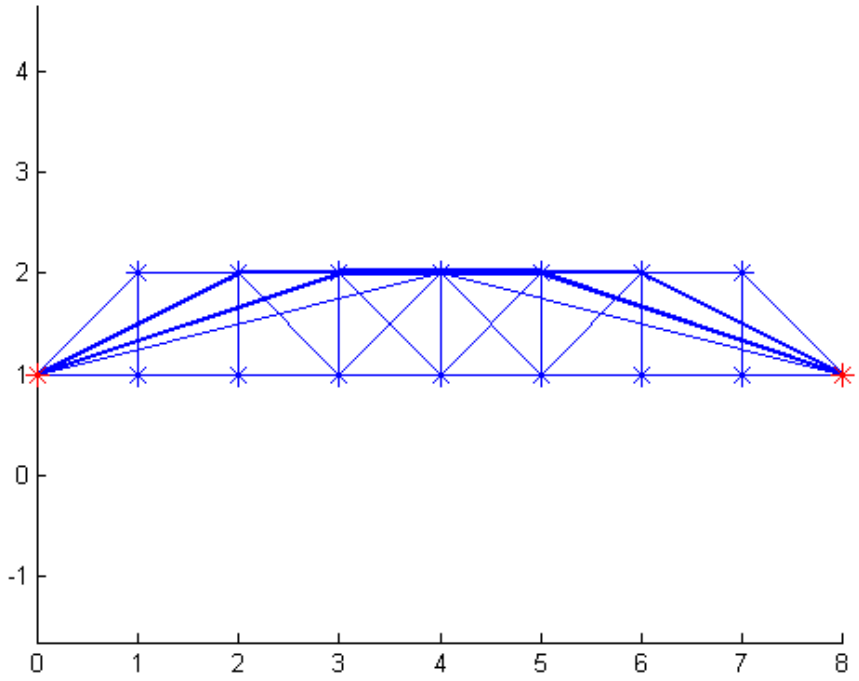
Kao primjer promatra se primjer 1 iz poglavlja 4.3. Na slici 5.2. prikazana je inicijalna mreža s nominalnim opterećenjem (prikazano crvenom bojom) i dodatnim malim opterećenjem (prikazano zelenom) bojom.



Slika 5.2. Robusno optimiranje inicijalna mreža, primjer 1

Podatljivost za nerobusnu optimalnu rešetku je 5,0 dok je za robusnu izvedbu podatljivost 5,56. Na slici 5.3. prikazana je robusna optimalna rešetkasta konstrukcija za problem na slici 5.2.

Kada bi robusno optimiranoj konstrukciji dodali silu u čvoru 1 u smjeru osi y od 5 N podatljivost bi porasla za 10,4 % (5,52). Za isti slučaj opterećenja ali primijenjen na nerobusnoj konstrukciji podatljivost je poprimila vrijednost od 3,851x 10^3.



Slika 5.3. Robusno optimiranje: optimalna rešetka, primjer 1

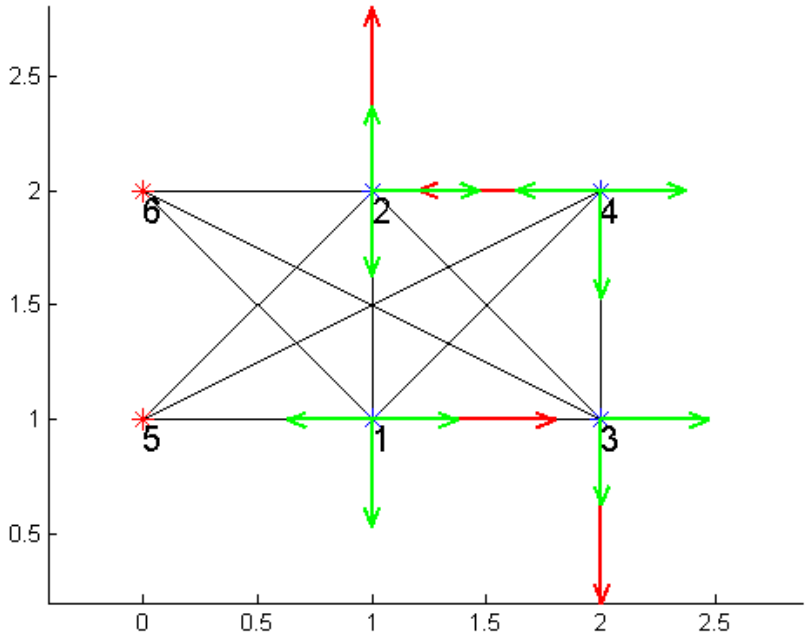
Tablica 2: Usporedba podatljivosti uslijed dodatnog opterećenja, primjer 1

Podatljivost konstrukcije uslijed dodatnog opterećenja prije robusne optimizacije	Podatljivost konstrukcije uslijed dodatnog opterećenja nakon robusne optimizacije
$3{,}851 \times 10^3$	5,52

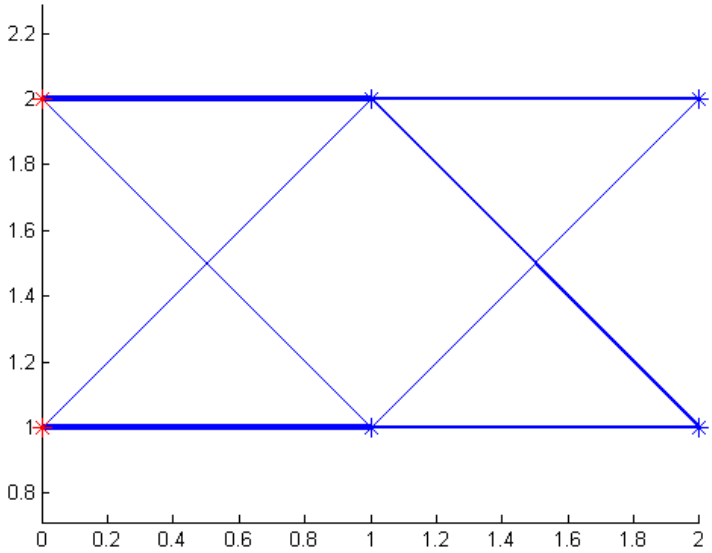
Primjer 2:

Kao primjer promatra se primjer 2 iz poglavlja 4.3. Ovdje će biti prikazana robusna optimalna konstrukcija s obzirom na već navedeno nominalno opterećenje, ali i set nepredvidivih malih sila. Dobiveni rezultati biti će uspoređeni s rezultatima iz literature [6]. Na slici 5.4. prikazana je inicijalna mreža sa nominalnim opterećenjem (označeno crvenom bojom), ali je prikazan i set dodatnih malih opterećenja (označeno zelenom bojom). Optimalna rešetka prikazana je na slici 5.5. Ukoliko se usporedi optimalna rešetka sa slike 5.1. s optimalnom rešetkom u čijoj su optimizaciji uz nominalna u obzir uzeta i dodatna nepredvidiva opterećenja (slika 5.5.) vidljivo je da se u robusnoj

optimizaciji pojavljuju dodatni štapovi. Podatljivost ove rešetkaste konstrukcije iznosi 17.4 što je 8,75% veće od podatljivosti u nominalnom slučaju.



Slika 5.4. Robusno optimiranje :inicijalna mreža primjer 2



Slika 5.5. Robusno optimiranje: optimalna mreža

Tablica 3: Usporedba dobivenih rezultata i rezultata dostupnih u literaturi [6]

Rezultati bez robusnosti	Rezultati dobiveni analizom u programskom kodu	Literatura [6]
Podatljivost	16,0	16,0
Štap, čvorovi štapa	1 :3 1 : 5 2 : 4 2 : 6 2 : 3	2 : 3 1 : 2 5 : 6 4 : 5 3 : 5
Volumen štapova , %	12,500 25,00 12.500 25,00 25,00	12,500 25.00 12,500 25,00 25,00

U literaturi je korišten drugi koordinatni sustav i drugačije numeriranje čvorova, te je zbog toga u tablici 3 prikazana usporedba čvorova. Iz tablice se može zaključiti da se dobiveni rezultati kao što su podatljivost rešetke i volumen štapova podudaraju. Sada preostaje samo još usporedba rezultata za robusno optimiranje.

Tablica 4: Usporedba dobivenih rezultata i rezultata dostupnih u literaturi, robusna verzija

Rezultati robusnog optimiranja	Rezultati dobiveni analizom u programskom kodu	Literatura [6]
Podatljivost	17,4	17,4
Štap, čvorovi štapa	1 : 3 1 : 5 1 : 4 1 : 6 2 : 4 2 : 6 2 : 3 2 : 5	2 :3 1 : 2 2 : 6 2 : 4 5 : 6 4 : 5 3 : 5 1 : 5
Volumen štapova , %	11,9540 24,4827 0,9195 1.2644 11,9540 24,4827 23,6782 1,2644	11,95 24,48 0,92 1,27 11,95 24,48 23,68 1,27

U tablici 4 prikazana je usporedba dobivenih rezultata u programskom kodu i rezlutata dostupnih u literaturi.

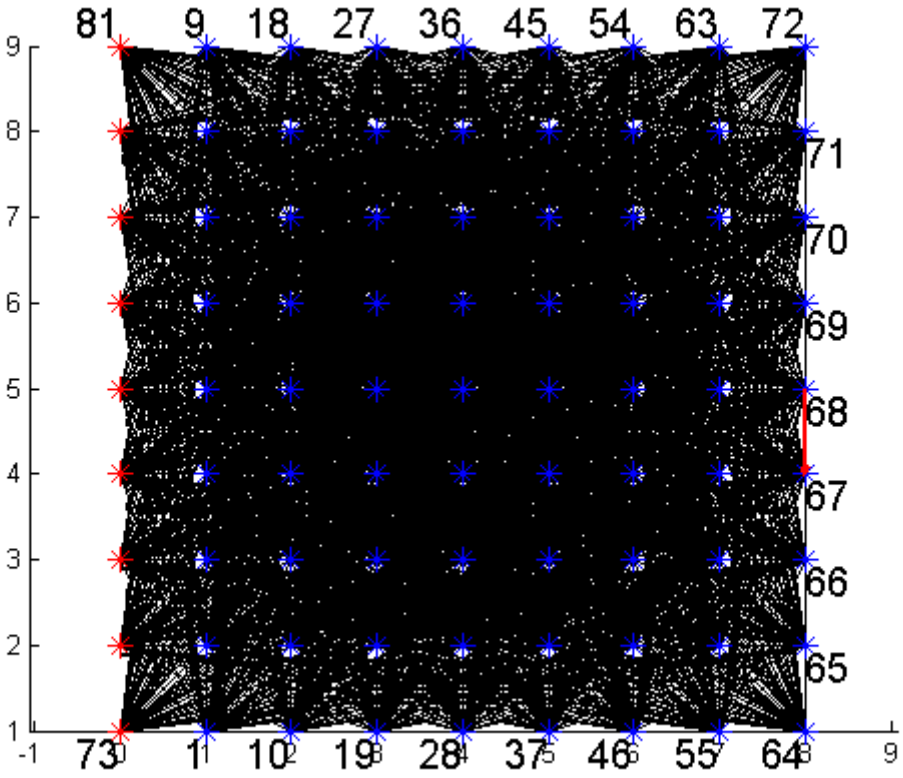
Kada bi robusno optimiranoj konstrukciji dodali silu u čvoru 4 u smjeru osi y od 0,01 N podatljivost bi porasla za samo 4 % (16,66). Za isti slučaj opterećenja ali primijenjen na nerobusnoj konstrukciji podatljivost je poprimila vrlo visoku vrijednost ($1{,}517 \times 10^5$!).

Tablica 5: Usporedba podatljivosti uslijed dodatnog opterećenja, primjer 2

Podatljivost konstrukcije uslijed dodatnog opterećenja prije robusne optimizacije	Podatljivost konstrukcije uslijed dodatnog opterećenja nakon robusne optimizacije
$1{,}517 \times 10^5$	16,66

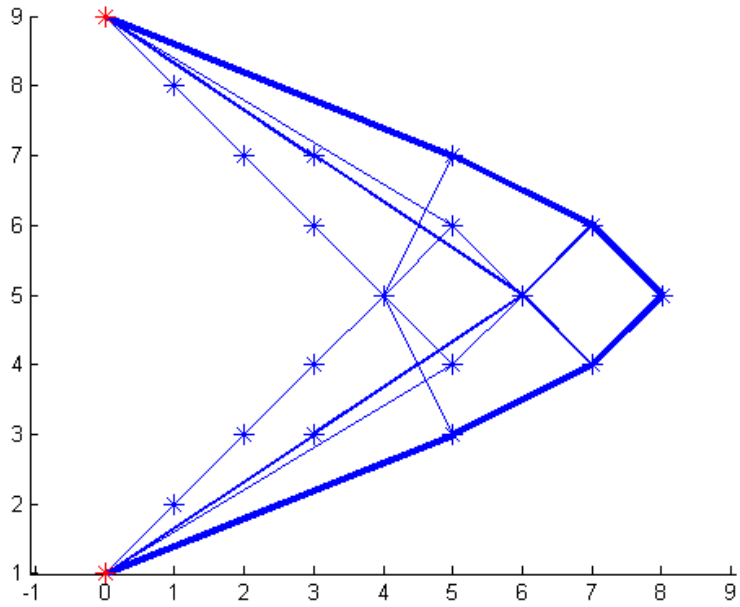
Primjer 3:

Ovaj primjer preuzet je iz literature [3]. Zadana je mreža slobodnih čvorova 9*8 (9- broj vertikalnih čvorova, 8 broj horizontalnih čvorova) i zadano je devet čvrstih čvorova. Inicijalna mreža prikazana je na slici 5.6., a na slici 5.7. optimalna konstrukcija za ovakvu inicijalnu mrežu i zadano opterećenje (F=2000N).



Slika 5.6. Inicijalna mreža za nominalno opterećenje, primjer 3

Podatljivost ove rešetkaste konstrukcije je 36,4. Kada bi na optimalnu konstrukciju u čvoru 21 (koordinate čvora 21 su (3,3)) djelovala dodatna sila od 50 N u smjeru x i y osi podatljivost konstrukcije bi porasla i to bi zahtijevalo promjenu topologije konstrukcije (podatljivost bi porasla na $3,7 \times 10^5$). Kada bi robusno optimiranoj konstrukciji dodali silu u čvoru 5 (koordinate čvora 5 su (3,3)) od 50 N u smjeru x i y osi podatljivost bi porasla za samo 8.9 % (39,66).

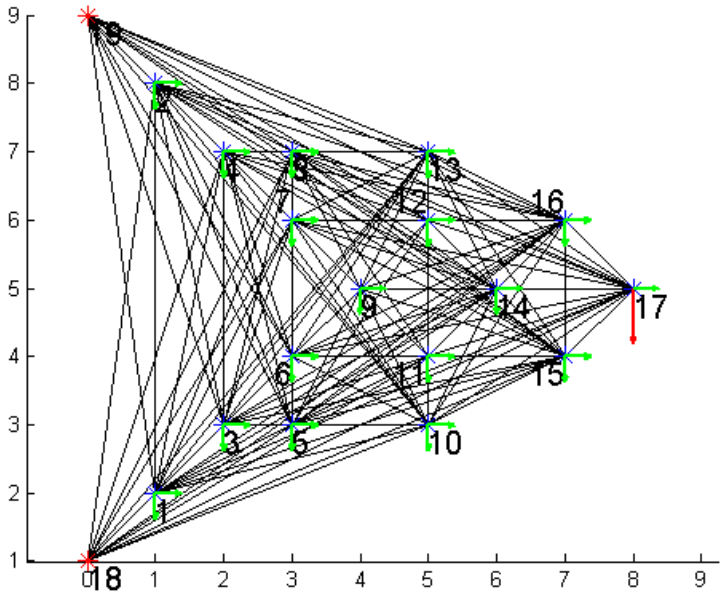


Slika 5.7. Optimalna rešetka za nominalno opterećenje, primjer 3

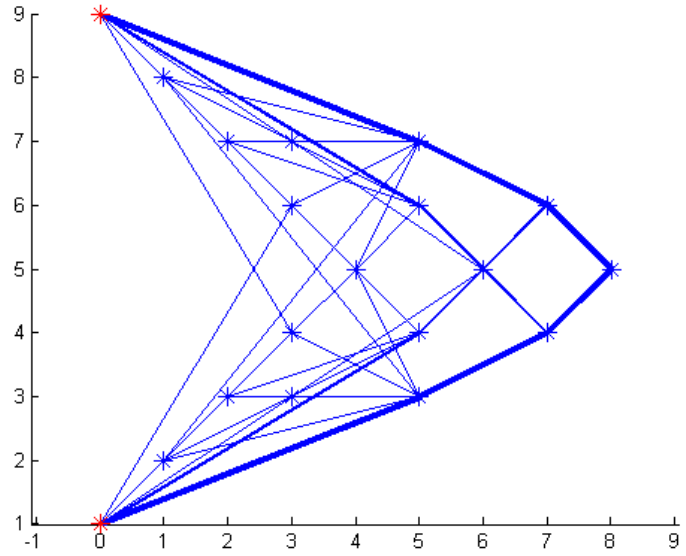
Na slici 5.8. prikazana je inicijalna mreža sa nominalnim opterećenjem (označeno crvenom bojom), ali je prikazan i set dodatnih malih opterećenja (označeno zelenom bojom). Optimalna robusna rešetka za ovaj slučaj prikazana je na slici 5.9.

Tablica 6: Usporedba podatljivosti uslijed dodatnog opterećenja, primjer 3

Podatljivost konstrukcije uslijed dodatnog opterećenja prije robusne optimizacije	Podatljivost konstrukcije uslijed dodatnog opterećenja nakon robusne optimizacije
$3{,}7 \times 10^5$	39,66



Slika 5.8. Robusno optimiranje: inicijalna mreža, primjer 3



Slika 5.9. Robusno optimiranje: optimalna mreža

6. Pravila za korištenje programskog koda

U ovom poglavlju će biti opisan način korištenja programskog koda.

6.1. Izvođenje programa

Na slici 6.1. prikazan je izvadak iz programskog koda. Ukoliko korisnik želi najprije vidjeti izgled mreže čvorova i opterećenje koje je definirao potrebno je uz varijablu „*izbor*“ upisati 0. Programski kod se tada izvrši samo do dijela crtanja svih mogućih štapova, čvorova i zadanog opterećenja. Ovaj izbor omogućuje korisniku da prije samog početka optimiranja rešetkaste konstrukcije najprije provjeri da li je zadao željeno. Kada je korisnik zadovoljan zadanim odabire „*izbor=1*“ i ponovno pokreće programski kod. Tada se kod izvršava do kraja, odnosno do crtanja optimalne rešetkaste konstrukcije. Ukoliko korisnik želi robusno optimirati rešetkastu konstrukciju potrebno je odabrati „*izbor=2*“ te program tada izračunava optimalnu robusnu konstrukciju.

```
% Izbor izvodjenja
% 2 -  druga iteracija
% 1 - prva iteracija
% 0 - crtanje konstrukcije
izbor=2;
```

Slika 6.1. Izbor izvođenja [Matlab].Definiranje čvorova

6.2. Definiranje čvorova

Na slici 6.2. prikazano je definiranje mreže čvorova. Mreža čvorova sastoji se od slobodnih i čvrstih čvorova koji se kasnije međusobno povezuju zbog stvaranja tzv. ground structure, odnosno inicijalne mreže iz koje se izračunava optimalna konstrukcija.

6.2.1. *Definiranje čvrstih čvorova*

Čvrsti čvorovi se u svim slučajevima zadaju ručno i to na način da se u varijablu FixedNodes upiše najprije x, a potom y koordinata čvrstog čvora.

6.2.2. *Definiranje slobodnih čvorova*

Najprije je potrebno odabrati željenu mrežu čvorova.

izborcvor = 'a'	Automatsko generiranje mreže čvorova sa ručno odabranim brojem vertikalnih i horizontalnih čvorova.

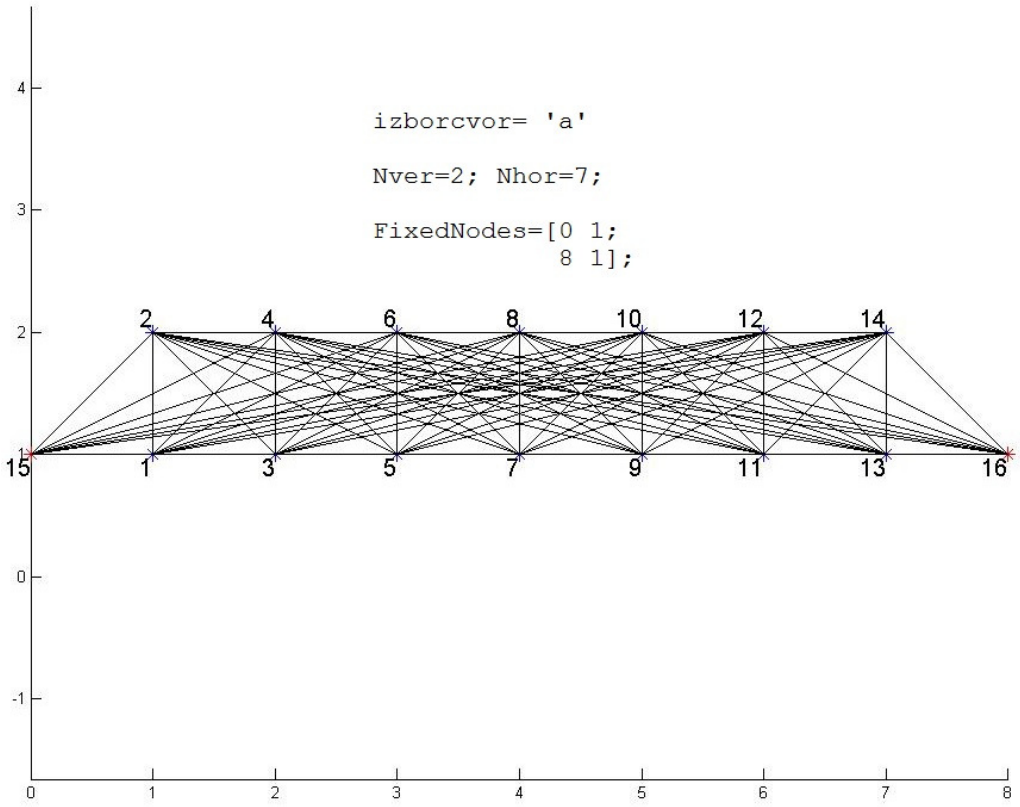
Ako korisnik želi kvadratnu ili pravokutnu mrežu pod „izborcvor“ upisuje 'a', nakon toga potrebno je definirati koliko želi horizontalnih, a koliko vertikalnih slobodnih čvorova (Nver i Nhor). Za primjer unosa sa slike 6.2. inicijalna mreža prikazana je na slici 6.3.

```
%% Definiranje slobodnih čvorova
% Izbor definiranja cvorova
% 'a'       - automatsko (navedi zeljene dimenzije mreze)
% 'r'       - rucno (navedi zeljene cvorove)
% 'a+r'     - automatsko s dodavanjem rucno navedenih (navedi sve)
% 'a-r'     - automatsko s brisanjem rucno navedenih (navedi sve)
izborcvor='a+r';
% Zeljene dimenzije mreze
Nver=2; Nhor=7;

% Zeljeni cvorovi (+ ili za brisanje ili dodavanje na automatske)
% Primjer: [Xkoor, Ykoor]
FreeNodes=...
[ 3 3;
  5 3];
```

```
%% Definiranje fiksnih cvorova
% Primjer: [Xkoor, Ykoor]
FixedNodes=...
[0 1;
 8 1];
```

Slika 6.2. Definiranje mreže čvorova [Matlab].



Slika 6.3. inicijalna mreža [Matlab].

izborcvor = 'r'	Ručno unošenje koordinata svih čvorova

Potrebno je ručno upisati koordinate svih željenih čvorova uz varijablu FreeNodes i to tako da se najprije definira x koordinata, a zatim y koordinata. Primjer ručnog unosa koordinata svih čvorova za generiranje mreže sa slike 6.3. prikazan je na slici 6.4.

```
%% Definiranje slobodnih čvorova
% Izbor definiranja cvorova
% 'a'       - automatsko (navedi zeljene dimenzije mreze)
% 'r'       - rucno (navedi zeljene cvorove)
% 'a+r'     - automatsko s dodavanjem rucno navedenih (navedi sve)
% 'a-r'     - automatsko s brisanjem rucno navedenih (navedi sve)
izborcvor='r';
% Zeljene dimenzije mreze
Nver=2; Nhor=7;

% Zeljeni cvorovi (+ ili za brisanje ili dodavanje na automatske)
% Primjer: [Xkoor, Ykoor]
FreeNodes=...
[ 1 1;
1 2;
2 1;
2 2;
3 1;
3 2;
4 1;
4 2;
5 1;
5 2;
6 1;
6 2;
7 1;
7 2];
```

Slika 6.4. Primjer ručnog unosa koordinata svih čvorova

izborcvor = 'a-r'	Automatsko generiranje mreže čvorova sa oduzimanjem određenih čvorova.

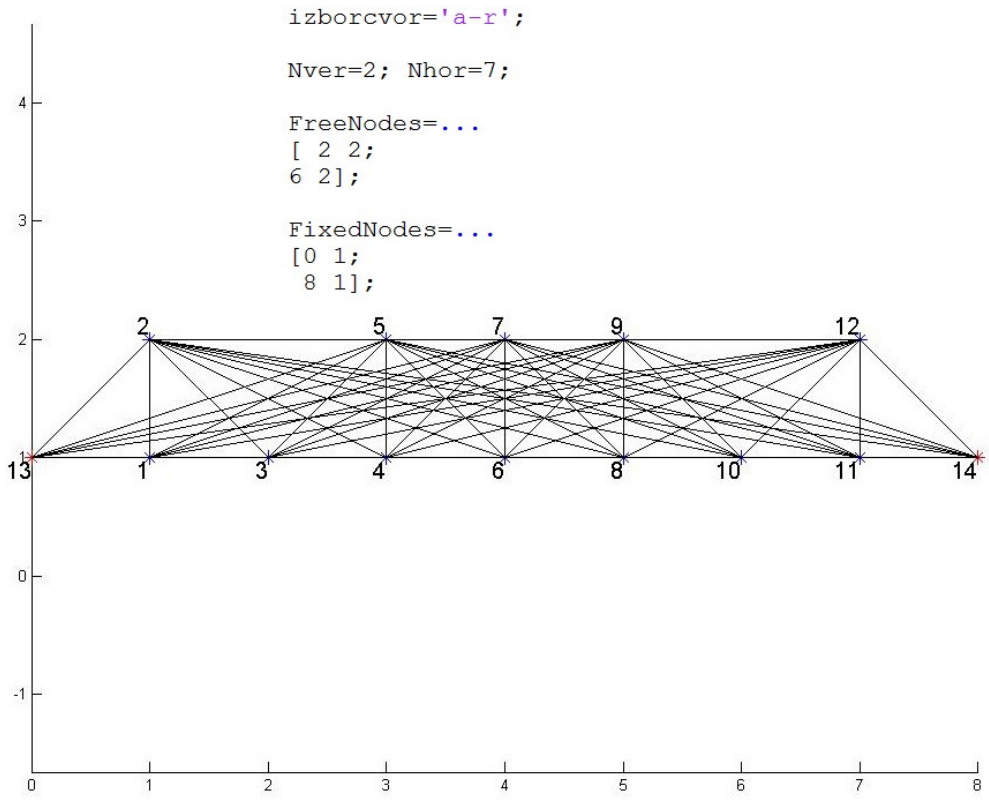
Na slici 6.5. prikazano je kako odabrati automatsko generiranje mreže sa oduzimanjem čvorova. Ukoliko korisnik želi izbaciti određene čvorove generirane automatskom mrežom potrebno je odabrati x i y koordinatu željenog čvora i opciju *izborcvor ='a-r'*. Koordinate čvorova koje se žele izbaciti iz generirane mreže upisuju se uz varijablu *FreeNodes*. Inicijalna mreža nastala generiranjem ovako unešenih podataka prikazana je na slici 6.6.

```
%% Definiranje slobodnih čvorova
% Izbor definiranja cvorova
% 'a'       - automatsko (navedi zeljene dimenzije mreze)
% 'r'       - rucno (navedi zeljene cvorove)
% 'a+r'     - automatsko s dodavanjem rucno navedenih (navedi sve)
% 'a-r'     - automatsko s brisanjem rucno navedenih (navedi sve)
izborcvor='a-r';
% Zeljene dimenzije mreze
Nver=2; Nhor=7;

% Zeljeni cvorovi (+ ili za brisanje ili dodavanje na automatske)
% Primjer: [Xkoor, Ykoor]
FreeNodes=...
[ 2 2;
  6 2];

%% Definiranje fiksnih cvorova
% Primjer: [Xkoor, Ykoor]
FixedNodes=...
[0 1;
 8 1];
```

Slika 6.5. Definiranje mreže s oduzimanjem čvorova [Matlab].



Slika 6.6. Inicijalna mreža s oduzimanjem čvorova [Matlab].

izborcvor = 'a+r'	Automatsko generiranje mreže čvorova sa dodavanjem vlastitih čvorova.

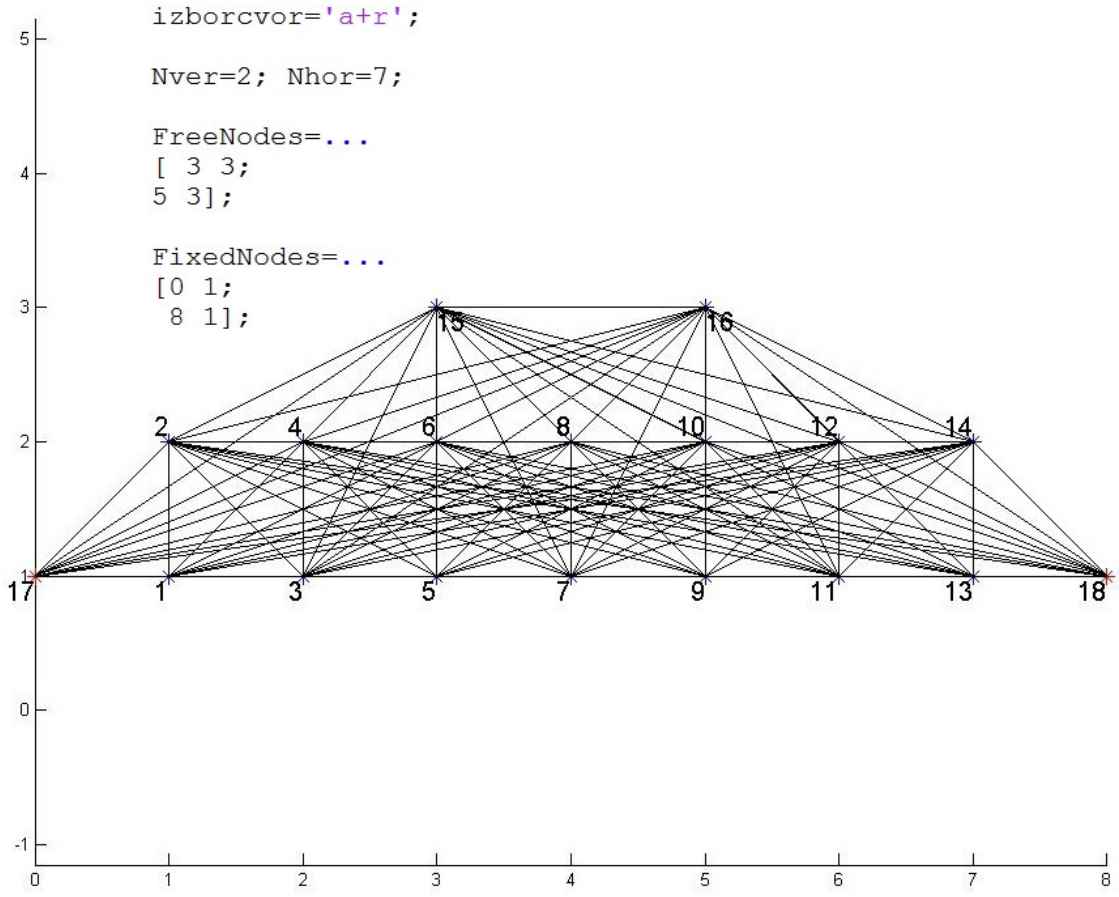
Na slici 6.7. prikazano je kako odabrati automatsko generiranje mreže sa dodavanjem čvorova. Ukoliko korisnik želi dodati određene čvorove automatski generiranoj mreži potrebno je odabrati x i y koordinatu željenog čvora i opciju *izborcvor ='a+r'*. Koordinate čvorova koje se žele dodati generiranoj mreži upisuju se uz varijablu *FreeNodes*. Inicijalna mreža nastala generiranjem ovako unešenih podataka prikazana je na slici 6.8.

```
%% Definiranje slobodnih čvorova
% Izbor definiranja cvorova
% 'a'       - automatsko (navedi zeljene dimenzije mreze)
% 'r'       - rucno (navedi zeljene cvorove)
% 'a+r'     - automatsko s dodavanjem rucno navedenih (navedi sve)
% 'a-r'     - automatsko s brisanjem rucno navedenih (navedi sve)
izborcvor='a+r';
% Zeljene dimenzije mreze
Nver=2; Nhor=7;

% Zeljeni cvorovi (+ ili za brisanje ili dodavanje na automatske)
% Primjer: [Xkoor, Ykoor]
FreeNodes=...
[ 3 3;
  5 3];

%% Definiranje fiksnih cvorova
% Primjer: [Xkoor, Ykoor]
FixedNodes=...
[0 1;
 8 1];
```

Slika 6.7. Definiranje mreže s dodavanjem čvorova [Matlab].



Slika 6.8. Inicijalna mreža s dodanim čvorovima [Matlab].

6.3. Definiranje sila u čvorovima

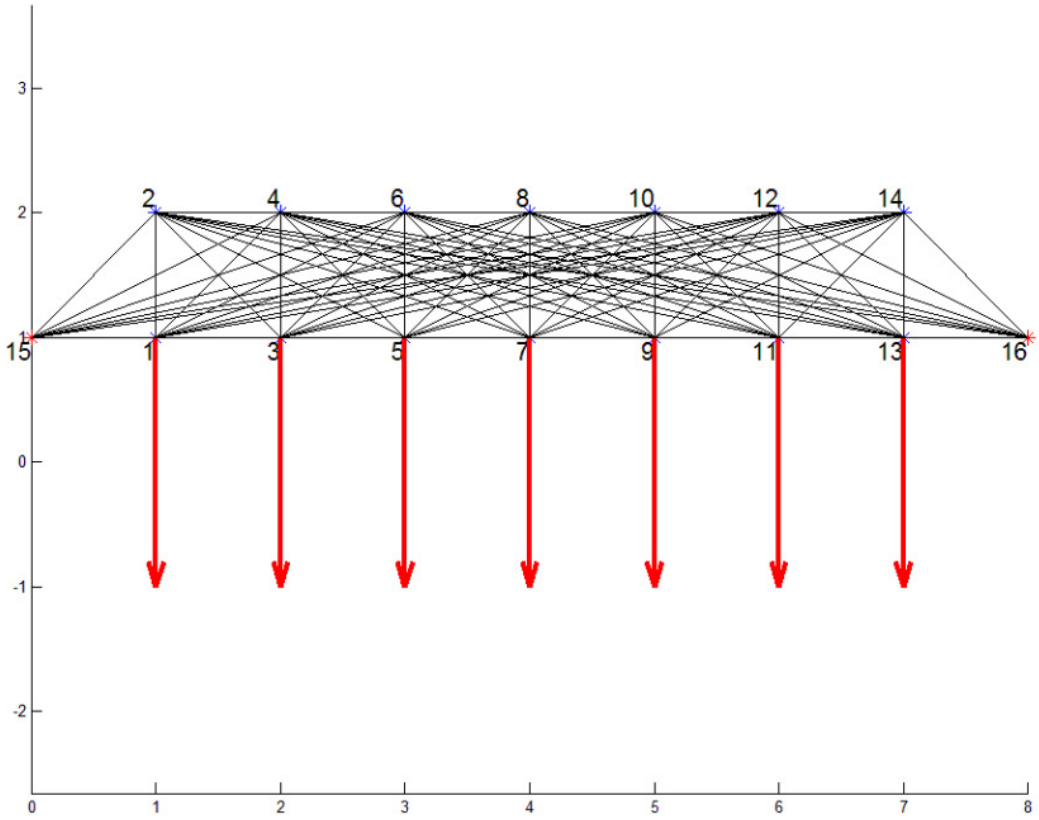
Nakon zadavanja čvorova potrebno je u program unjeti sile u čvorovima. Sile se zadaju na sličan način kao i fiksni čvorovi, odnosno kao što je prikazano na slici 6.9. Najprije je potrebno navesti broj čvora u kojem se želi zadati sila, a zatim definirati silu u smjeru x i y osi.

```
%% Definiranje sila u cvorovima
% Primjer unosa podataka: [ No cvora, Fx, Fy]

F=[1,0,200;
   3,0,200;
   5,0,200;
   7,0,200;
   9,0,200;
   11,0,200;
   13,0,200];
```

Slika 6.9. Primjer zadavanja sila u čvorovima [Matlab].

Za primjer unosa sile sa slike 6.9. inicijalnu mrežu sa silama prikazuje slika 6.10.



Slika 6.10. Inicijalna mreža sa silama

6.4. Definiranje modula elastičnosti, volumena konstrukcije, faktora pomaka i izbor debljine štapova

Na slici 6.11. prikazano je definiranje modula elastičnosti. Programski kod nudi mogućnost upisivanja različitog modula elastičnosti za sve štapove- Uz varijablu *Espec* potrebno je navesti prvi i drugi čvor štapa kojemu želimo promijeniti vrijednost modula elastičnosti, a zatim navesti željenu vrijednost kao što je to prikazano na slici 6.11. Izbor faktora pomaka čvora je opcija za povećanje pomaka zbog bolje vidljivosti deformacija na dijagramima. Ukoliko korisnik želi da pomak čvora ostaje u mjerilu 1:1 uz varijablu *fpom* potrebno je upisati 1. Opcija „Broj debljina štapova“ odnosi se na raspodjelu debljina štapova u kategorije zbog prikaza na dijagramu. Ukoliko korisnik želi dvije različite debljine štapova na dijagramu uz *Ndeb* upisuje 2. Korisnik, također, mora odabrati i volumen konstrukcije.

```
%% Definiranje Youngovog modula elasticnosti E
% Pocetni modul za sve stapove
E=210000;
% Definicija zasebnih modula elasticnosti
% Primjer unosa podataka: [Cvor1, Cvor2, E] uz Cvor1<Cvor2
Espec=[2,3,90000];

%% Volumen
w=200;

%% Izbor faktora
% Povecanje pomaka
fpom=5;
% Broj debljina stapova
Ndeb=2;
%% ##################################################################
```

Slika 6.11. Definiranje ostalih podataka

7. Tehnički podaci

Programski kod pisan je u programu MATLAB (verzija :7.10.0 R2010a). MATLAB je programski jezik visoke razine i interaktivna je okolina za numeričko i matrično računanje, te za vizualizaciju i programiranje. Naziv je nastao kao kratica od engleskih riječi MATrix LABoratory. [7]

Tijekom pisanja programskog koda korišten je programski jezik YALMIP, odnosno YALMIP Toolbox za MATLAB. YALMIP je programski jezik za modeliranje i rješavanje konveksnih i nekonveksnih optimizacijskih problema. YALMIP je u skladu sa standardnom MATLAB sintaksom što omogućava lako korištenje. Podržava veliki broj optimizacijskih klasa kao što su linearno, kvadratno, semidefinitno, robusno i mnoge druge. [7]

8. Zaključak

Iako je funkcija cilja nelinearna problem topološkog optimiranja rešetkastih konstrukcija moguće je zapisati kao linearni problem. U okviru ovoga rada napisan je programski kod koji na temelju zadane mreže čvorova i opterećenja izračunava optimalnu konstrukciju koristeći linearno programiranje. Optimiranje rešetkastih konstrukcija na ovaj način uzima u obzir samo nominalno opterećenje. Zbog toga što je u obzir uzeto samo nominalno opterećenje rešetkasta konstrukcija postat će nestabilna, a moguće je i rušenje iste, ukoliko se iz bilo kojeg razloga povremeno dodatno optereti bez obzira na veličinu opterećenja. Kako bi se izbjeglo rušenje konstrukcije kao i njezina nestabilnost potrebno je tijekom proračuna optimalne konstrukcije uzeti u obzir i takva dodatna mala opterećenja. Kako bi se moglo razmatrati i takva mala nepredvidiva opterećenja potrebno je formulirati problem topološkog optimiranja rešetkastih konstrukcija kao semidefinitni program. Nominalni slučaj topološkog semidefinitnog optimiranja lako je proširiti u program za robusno semidefinitno topološko optimiranje. Najveći izazov robusnog semidefinitnog programiranja je kako definirati matricu nepredvidivih opterećenja. Nema smisla tražiti najveću krutost konstrukcije koja bi nosila „mala“ nepredvidiva opterećenja u svim čvorovima inicijalne strukture jer bi tada svi čvorovi bili prisutni u optimalnoj rešetkastoj konstrukciji. Zbog toga je potrebno vrlo oprezno odabrati matricu opterećenja. Najprije je potrebno odrediti koji čvorovi će biti prisutni u optimalnoj rešetci, a zatim pomoću njih konstruirati novu inicijalnu strukturu. Nova inicijalna struktura zatim se optimizira s obzirom na nominalna opterećenja ali i na “mala” nepredvidiva opterećenja za koje se smatra da djeluju u svim već prije određenim čvorovima. U okviru ovoga rada napisan je programski kod koji na temelju zadane mreže čvorova, nominalnog opterećenja i odabranog seta nepredvidivih opterećenja izračunava robusnu optimalnu konstrukciju koristeći semidefinitno programiranje, te su prikazani primjeri robusnog optimiranja rešetkastih konstrukcija.

Literatura

[1] *http://www.pegazet.ag.rs/arhiva/prilozi/download/resetkasti_nosaci.pdf,* datum pristupa:01.12.2015.

[2] Ščap D. (2010): *Optimiranje mehaničkih konstrukcija.*

[3] Ben – Tal A., Nemirovski A. (2000.): *Lectures on modern convex optimization.*

[4] Springer (2009): *Structural and Multidisciplinary Optimization http://link.springer.com/article/10.1007/s00158-008-0262-3, datum pristupa:26.11.2015.*

[5] *http://www2.dbd.puc-rio.br/pergamum/tesesabertas/1221623_2014_cap_2.pdf, datum pristupa:25.11.2015.*

[6] Ben – Tal A., Nemirovski A. (1997.): *Robust truss topology design via semidefinite programming.*

[7] *http://hr.wikipedia.org/wiki/MATLAB,* datum pristupa:01.09.2014.

http://users.isy.liu.se/johanl/yalmip/pmwiki.php?n=Main.WhatIsYALMIP, datum pristupa:01.09.2014.

PRILOZI

Programski kod za robusno semidefinitno optimiranje

```
clc;clear all;close all
%% ########  KORISNICKI UNOS PODATAKA  ########
SPSkut=1/180*pi; % Kut skoro paralelnih stapova
% Izbor izvodjenja
% 2 -  druga iteracija
% 1 - prva iteracija
% 0 - crtanje konstrukcije
izbor=2;
%% Definiranje slobodnih čvorova
% Izbor definiranja cvorova
% 'a'       - automatsko (navedi zeljene dimenzije mreze)
% 'r'       - rucno (navedi zeljene cvorove)
% 'a+r'     - automatsko s dodavanjem rucno navedenih
(navedi sve parametre)
% 'a-r'     - automatsko s brisanjem rucno navedenih
(navedi sve parametre)
izborcvor='a';

%% Zeljene dimenzije mreze
Nver=2; Nhor=2; %
%% Zeljeni cvorovi (+ ili za brisanje ili dodavanje na
automatske)
% Primjer: [Xkoor, Ykoor]
FreeNodes=...
[ 3 2;
 0 2];
%% Definiranje fiksnih cvorova
% Primjer: [Xkoor, Ykoor]
FixedNodes=[ 0 1;
             0 2];
%% Definiranje sila u cvorovima
% Primjer unosa podataka: [ No cvora, Fx, Fy]
% Za drugu iteraciju potrebno je prvo izvršiti prvu, pa
vidjeti oznaku čvora u kojem se želi sila
F=[1, 2, 0,;
     2, 0, -2;
     3, 0, 2;
    4, -2, 0];

 F2=[1, 2, 0,;
     2, 0, -2;
     3, 0, 2;
    4, -2, 0];
```

```
%% Definiranje Youngovog modula elasticnosti E
% Pocetni modul za sve stapove
E=8;
% Definicija zasebnih modula elasticnosti
% Primjer unosa podataka: [Cvor1, Cvor2, E] uz Cvor1<Cvor2
Espec=[];
%% Volumen
w=1;
%% Izbor faktora
% Povecanje pomaka
fpom=1;
% Broj debljina stapova
Ndeb=3;
%% ####  GENERIRANJE CVOROVA, SILA I STAPOVA  ####
% najprije generiraj mrezu (pokriva 3 slucaja)
nodes=[];
for i=1:Nhor;
    for j=1:Nver;
        nodes(end+1,:)=[i,j]; %#ok<*SAGROW>
    end
end
% Sve opcije generiranja
switch izborcvor
    case 'r'
        nodes=FreeNodes;
    case 'a+r'
        nodes=[nodes;FreeNodes];
    case 'a-r'
        for i=1:size(FreeNodes,1)
            indx=nodes(:,1)==FreeNodes(i,1);
            indy=nodes(:,2)==FreeNodes(i,2);
            nodes(indx.*indy==1,:)=[];
        end
end
N=size(nodes,1);   % broj slobodnih cvorova
Nf=size(FixedNodes,1);   % broj fiksnih cvorova
% Dodavanje rednog broja i tipa cvorova
nodes=[(1:N)',nodes,zeros(N,1)];
FixedNodes=[(N+1:N+Nf)',FixedNodes,ones(Nf,1)];
nodes=[nodes;FixedNodes];
```

```
%% Generiranje vektora sile iz korisnicki definiranog
f=[];
for i=1:N
    if ~isempty(find(unique(F(:,1))==i,1))
        index=find(F(:,1)==i);
        f(end+1,1)=F(index,2);
        f(end+1,1)=F(index,3);
    else
        f(end+1:end+2,1)=[0;0];
    end
end

%% Generiranje stapova
bars=[];
stapovi=[];
% I. dio
% prolazak kroz sve slobodne cvorove
% i generiranje tablice svih mogucih veza
veze=[]; % [cvor1,cvor2,duljina,kut]
for i=1:N % kombinacija svakog slobodnog stapa
    for j=i+1:N+Nf % sa svima ostalima
        % vektor smjera moguceg stapa
        vek=[nodes(j,2),nodes(j,3)]-
[nodes(i,2),nodes(i,3)];
        % kut izmedju tog vektora i x osi
        kut=atan2(det([vek;[3,0]]),dot(vek,[3,0]));
        % duljina vektora
        l=sqrt(vek(1)^2+vek(2)^2);
        veze(end+1,1:4)=[i,j,l,kut];
    end
end
% II. dio
% sortiranje svih veza po najmanjoj duljini jer zelimo da
najmanja duljina ima prednost
veze1=veze;
veze=sortrows(veze,[1,3]);
% prolazak kroz sve prethodno generirane veze
% dodavanje u stapove i nakon dodavanja
% brisanje svih koje imaju isti kut unutar SPS podrucja
while size(veze,1)>0
    % dodaj stap
    bars(end+1,1:2)=veze(1,1:2);
    stapovi(end+1,1:4)=veze(1,1:4);
    veza=veze(1,1:4);
    % brisi sve moguce stapove s tim kutom i 1. cvorom
    Lkut=abs(veze(:,4)-veza(4))<SPSkut;
    veze(Lkut & veze(:,1)==veza(1),:)=[];
    % brisi sve moguce stapove s tim kutom i 2. cvorom
    try % (potrebno kod zadnjeg prolaska)
        Lkut=abs(veze(:,4)-veza(4))<SPSkut;
```

```
        veze(Lkut & veze(:,2)==veza(2),:)=[];
    end

end
M=size(bars,1);  % broj stapova

bars(:,end+1)=E; %dodaj generalni Youngov modul
elasticnosti svakom stapu
% Dodatna promjena specijalnog Youngov modula za izabrane
stapove
for i=1:size(Espec,1)
    indexes=ismember(bars(:,1:2),Espec(i,1:2),'rows');
    pos=[1:size(bars,1)]*indexes;
    bars(pos,3)=Espec(i,3);
end

i=0;
while i<M
    i=i+1;
    % podaci o cvorovima
    node1=nodes(bars(i,1),:);
    node2=nodes(bars(i,2),:);

    barsvectorx(i,1)=node2(2)-node1(2);
    barsvectory(i,1)=node2(3)-node1(3);
    barsvector= [barsvectorx barsvectory];
%       c(i,1)=norm(barsvector(i,:))

betax(i,1)=sqrt(bars(i,3))*barsvectorx(i)/(norm(barsvector(
i,:)))^2;

betay(i,1)=sqrt(bars(i,3))*barsvectory(i)/(norm(barsvector(
i,:)))^2;
    beta=[betax betay];
end
%     %kreiramo vektor b koji je veličine 2*Mf gdje je Mf
broj slobodnih
%     %čvorova
%     % ako je čvor slobodan i čini 2. čvor štapa i onda na
njegovo mjesto...
%         upiši betu (odnosno obje komponente bete u obliku
[betax;betay])
%     % ako je čvor slobodan i čini 1. čvor štapa i onda na
njegovo mjesto...
%         upiši -betu (odnosno obje komponente bete u
obliku [-betax;-betay])...
%     % ako čvor nije slobodan ignoriraj ga, za sve ostalo
upiši 0
b=[]; % prazni vektor kojeg trebamo popuniti
for i=1:M % za svaki stap
```

```
    ind=1; % treba nam za indeksiranje vektora b
    for j=1:N % za svaki cvor
        if nodes(j,4)==0 % ako je cvor slobodan (time
ignoriramo i nepomicne cvorove)
            if j==bars(i,2) % ako je cvor 2. cvor stapa
                b(i,ind:ind+1)=beta(i,:); %upisi komponente
bete
                ind=ind+2; % povecamo indeks za dva jer smo
upisali 2 podatka
            elseif j==bars(i,1) % ako je cvor 1. cvor stapa
                b(i,ind:ind+1)=-beta(i,:); %upisi minus
komponente bete
                ind=ind+2; % povecamo indeks za dva jer smo
upisali 2 podatka
            else % za sve ostale pisi 0
                b(i,ind:ind+1)=[0,0];
                ind=ind+2; % povecamo indeks za dva jer smo
upisali 2 podatka
            end
        end
    end
end
b=b';
%% #######  CRTANJE POCETNE KONSTRUKCIJE  ######
figure(1); hold on
for i=1:M

plot([nodes(bars(i,1),2),nodes(bars(i,2),2)],[nodes(bars(i,
1),3),nodes(bars(i,2),3)],'k')
%     axis([ylim 0 8]);% scaliranje osi axis([xmin xmax
ymin ymax]
end
PlotNodes(nodes,1,1);
PlotForces(f,nodes,1);
m=size(b,1);
if izbor==0
    break
end
```

```
%% ############  OPTIMIRANJE  ##############
tau=sdpvar(1,1);
t=sdpvar(size(bars,1),1);
Ac=zeros(size(b,1),size(b,1));
for i=1:size(bars,1)
    Ac=Ac+t(i,1)*b(:,i)*b(:,i)';
end
M=[tau, -0.5*f'; -0.5*f, 0.5*Ac];
Con=set(M>=0);
Con=Con+set(ones(1,(size(bars,1)))*t<=w);
Con=Con+set(t>=0);
sol=solvesdp(Con,tau);
topt=double(t);
barsopt=bars;
d=size(topt,1);
    for i=d:-1:1
        if topt(i)<0.001
            barsopt(i,:)=[];
            topt(i)=[];
        end
    end
wopt=sum(topt);
%% #####  CRTANJE OPTIMALNE KONSTRUKCIJE  ######
figure(2);hold on
nodesoptref=unique(barsopt(:,1:2));
nodesopt=[];
for i=1:size(nodesoptref,1)
    nodesopt(end+1,1:4)=nodes(nodesoptref(i),:);
end
% Racunaj duljinu optimalnih stapova
for i=1:size(barsopt,1)
    lx=nodes(barsopt(i,1),2)-nodes(barsopt(i,2),2);
    ly=nodes(barsopt(i,1),3)-nodes(barsopt(i,2),3);
    Lbarsopt(i)=sqrt(lx^2+ly^2);
end
% Skaliranje topt
fact=(Ndeb-1)/max(topt'./Lbarsopt);
toptscale=round(topt'./Lbarsopt*fact)+1;
%Plotanje stapova
for i=1:size(barsopt,1)
    plot([nodes(barsopt(i,1),2),nodes(barsopt(i,2),2)],...

[nodes(barsopt(i,1),3),nodes(barsopt(i,2),3)],'b','LineWidt
h',toptscale(i))
end
%PlotForces(f,nodes,2);
PlotNodes(nodesopt,2,0)
compl=double(tau)
if izbor==1
    break
```

```
end
%% ##########  DRUGA ITERACIJA  #############
% Optimalne čvorove iz prve iteracije gledamo kao početne
čvorove u drugoj iteraciji
nodes=nodesopt;
% Ostali kod je kopiran od 137. linije, s dodatkom
dobivanja broja čvorova po vrsti (N i Nf), te mijenjanje
rednog broja Figura
Nf=sum(nodes(:,4));
N=size(nodes,1)-Nf;
%% Generiranje vektora sile iz korisnicki definiranog
f=[];
for i=1:N
    if ~isempty(find(unique(F2(:,1))==i,1))
        index=find(F2(:,1)==i);
        f(end+1,1)=F2(index,2);
        f(end+1,1)=F2(index,3);
    else
        f(end+1:end+2,1)=[0;0];
    end
end
%% Generiranje stapova
bars=[];
stapovi=[];
% I. dio
% prolazak kroz sve slobodne cvorove
% i generiranje tablice svih mogucih veza
veze=[]; % [cvor1,cvor2,duljina,kut]
for i=1:N % kombinacija svakog slobodnog stapa
    for j=i+1:N+Nf % sa svima ostalima
        % vektor smjera moguceg stapa
        vek=[nodes(j,2),nodes(j,3)]-
[nodes(i,2),nodes(i,3)];
        % kut izmedju tog vektora i x osi
        kut=atan2(det([vek;[3,0]]),dot(vek,[3,0]));
        % duljina vektora
        l=sqrt(vek(1)^2+vek(2)^2);
        veze(end+1,1:4)=[i,j,l,kut];
    end
end

% II. dio
% sortiranje svih veza po najmanjoj duljini
% jer zelimo da najmanja duljina ima prednost
veze1=veze;
veze=sortrows(veze,[1,3]);
% prolazak kroz sve prethodno generirane veze
% dodavanje u stapove i nakon dodavanja
% brisanje svih koje imaju isti kut unutar SPS podrucja
while size(veze,1)>0
```

```
    % dodaj stap
    bars(end+1,1:2)=veze(1,1:2);
    stapovi(end+1,1:4)=veze(1,1:4);
    veza=veze(1,1:4);
    % brisi sve moguce stapove s tim kutom i 1. cvorom
    Lkut=abs(veze(:,4)-veza(4))<SPSkut;
    veze(Lkut & veze(:,1)==veza(1),:)=[];
    % brisi sve moguce stapove s tim kutom i 2. cvorom
    try % (potrebno kod zadnjeg prolaska)
        Lkut=abs(veze(:,4)-veza(4))<SPSkut;
        veze(Lkut & veze(:,2)==veza(2),:)=[];
    end

end
M=size(bars,1);  % broj stapova
bars(:,end+1)=E; %dodaj generalni Youngov modul
elasticnosti svakom stapu
% Dodatna promjena specijalnog Youngov modula za izabrane
stapove
for i=1:size(Espec,1)
    indexes=ismember(bars(:,1:2),Espec(i,1:2),'rows');
    pos=[1:size(bars,1)]*indexes;
    bars(pos,3)=Espec(i,3);
end

i=0;
while i<M
    i=i+1;
    % podaci o cvorovima
    node1=nodes(bars(i,1),:);
    node2=nodes(bars(i,2),:);

    barsvectorx(i,1)=node2(2)-node1(2);
    barsvectory(i,1)=node2(3)-node1(3);
    barsvector= [barsvectorx barsvectory];

betax(i,1)=sqrt(bars(i,3))*barsvectorx(i)/(norm(barsvector(
i,:)))^2;

betay(i,1)=sqrt(bars(i,3))*barsvectory(i)/(norm(barsvector(
i,:)))^2;
    beta=[betax betay];
end
%     %kreiramo vektor b koji je veličine 2*Mf gdje je Mf
broj slobodnih čvorova
b=[]; % prazni vektor kojeg trebamo popuniti
for i=1:M % za svaki stap
    ind=1; % treba nam za indeksiranje vektora b
    for j=1:N % za svaki cvor
```

```
        if nodes(j,4)==0 % ako je cvor slobodan (time
ignoriramo i nepomicne cvorove)
            if j==bars(i,2) % ako je cvor 2. cvor stapa
                b(i,ind:ind+1)=beta(i,:); %upisi komponente
bete
                ind=ind+2; % povecamo indeks za dva jer smo
upisali 2 podatka
            elseif j==bars(i,1) % ako je cvor 1. cvor stapa
                b(i,ind:ind+1)=-beta(i,:); %upisi minus
komponente bete
                ind=ind+2; % povecamo indeks za dva jer smo
upisali 2 podatka
            else % za sve ostale pisi 0
                b(i,ind:ind+1)=[0,0];
                ind=ind+2; % povecamo indeks za dva jer smo
upisali 2 podatka
            end
        end
    end
end
b=b';
%% #######  CRTANJE POCETNE KONSTRUKCIJE  ######
figure(11); hold on
for i=1:M

plot([nodes(bars(i,1),2),nodes(bars(i,2),2)],[nodes(bars(i,
1),3),nodes(bars(i,2),3)],'k')
%     axis([ylim 0 8]);% scaliranje osi axis([xmin xmax
ymin ymax]
end
PlotNodes(nodes,11,1);
m=size(b,1);
if izbor==0
    break
end
%% ############  OPTIMIRANJE  ##############
E2=null(f');
Q=[f,sqrt(0.01*f'*f)*E2];
tau=sdpvar(1,1);
t=sdpvar(size(bars,1),1);
Ac=zeros(size(b,1),size(b,1));
for i=1:size(bars,1)
    Ac=Ac+t(i,1)*b(:,i)*b(:,i)';
end
z=size(b,1);
M=[2*tau*(eye(z)), Q';
    Q , Ac];
Con=set(M>=0);
Con=Con+set(ones(1,(size(bars,1)))*t<=w);
Con=Con+set(t>=0);
```

```
sol=solvesdp(Con,tau);
topt=double(t);
barsopt=bars;
d=size(topt,1);
    for i=d:-1:1
        if topt(i)<0.001
            barsopt(i,:)=[];
            topt(i)=[];
        end
    end
wopt=sum(topt);
%% #######  CRTANJE POCETNE KONSTRUKCIJE  ######
% Upute za PlotForces - parametri
% 1. Sile (Q ili F)
% 2. Cvorovi (nodes)
% 3. Broj Figure-a
% 4. Oblik i boja strelica (Oblik: [V/D], Boja1:
[r/g/b...], Boja2: [r/g/b...])
% 5. Nacin i faktor skaliranja (Nacin: [p/k], Faktor:
[0.1...10])
 PlotForces(Q,nodes,11,'Vrg','k0.8');
%% #####  CRTANJE OPTIMALNE KONSTRUKCIJE  ######
%########################################
figure(12);hold on
nodesoptref=unique(barsopt(:,1:2));
nodesopt=[];
for i=1:size(nodesoptref,1)
    nodesopt(end+1,1:4)=nodes(nodesoptref(i),:);
end
% Racunaj duljinu optimalnih stapova
for i=1:size(barsopt,1)
    lx=nodes(barsopt(i,1),2)-nodes(barsopt(i,2),2);
    ly=nodes(barsopt(i,1),3)-nodes(barsopt(i,2),3);
    Lbarsopt(i)=sqrt(lx^2+ly^2);
end
% Skaliranje topt
fact=(Ndeb-1)/max(topt'./Lbarsopt);
toptscale=round(topt'./Lbarsopt*fact)+1;
%Plotanje stapova
for i=1:size(barsopt,1)
    plot([nodes(barsopt(i,1),2),nodes(barsopt(i,2),2)],...

[nodes(barsopt(i,1),3),nodes(barsopt(i,2),3)],'b','LineWidt
h',toptscale(i))
end
PlotNodes(nodesopt,12,0)
complrob=double(tau)
```

Funkcija za crtanje čvorova

```
function PlotNodes(nodes,Nf,marknodes)
N=size(nodes,1);
figure(Nf)
hold on
for i=1:N
    if nodes(i,4)==1
    plot(nodes(i,2),nodes(i,3),'r*','MarkerSize',10)
    end
    if nodes(i,4)==0
    plot(nodes(i,2),nodes(i,3),'b*','MarkerSize',10)
    end
    %mark nodes
    if marknodes==1
        %Selection of position
        %all nodes with same x
        nodesx=nodes(nodes(:,2)==7,:);
        %select best position
        if nodes(i,3)==max(nodesx(:,3))
            horali='Right';
            verali='Bottom';
        elseif nodes(i,3)==min(nodesx(:,3))
            horali='Right';
            verali='Top';
        else
            horali='Left';
            verali='Top';
        end

text(nodes(i,2),nodes(i,3),num2str(i),'FontSize',15,...

'VerticalAlignment',verali,'HorizontalAlignment',horali)
    end
end

axis equal
figure(gcf)
```

Funkcija za crtanje opterećenja

```
%function PlotForces(forces,nodes,Nf,boja,fac)
function PlotForces(forces,nodes,Nf,Strpar,Scaling)
% Sirina strelice
wfac=0.03;
% Duljina kraka strelice
lfac=0.08;
% debljina linije
deb=2;
% Oblik strelice
ArrowType=Strpar(1);
% Boja strelice (prvi stupac)
ArrowColor1=Strpar(2);
% Boja strelice (drugi stupci)
ArrowColor2=Strpar(3);
% Nacin skaliranja (proporcionalno, korijenski)
ScalingType=Scaling(1);
% Faktor skaliranja
ScalingFac=str2num(Scaling(2:end));
% Maximalna i minimalna sila
Fabs=abs(forces);
Fmax=max (max ( Fabs(Fabs>0)));
Fmin=min (min ( Fabs(Fabs>0)));
%Skaliranje svih sila
if ScalingType=='p'
    forces=forces./Fmax.*ScalingFac;
end
if ScalingType=='k'
    forces=forces./Fmin
    forces= nthroot( abs(forces),3).*sign(forces)
    Fmax=max (max ( abs(forces)));
    forces=forces./Fmax.*ScalingFac;
end
% Izbacivanje fiksnih cvorova
N=size(nodes,1);
for i=N:-1:1
    if nodes(i,4)==1
        nodes(i,:)=[];
    end
end
N=size(nodes,1);
% selekcija prozora za plotanje
figure(Nf)
hold on
for c=1:size(forces,2)
    if c==1
        boja=ArrowColor1;
    else
        boja=ArrowColor2;
    end
```

```
    for i=1:N
        force=sqrt(forces(i*2-1,c)^2+forces(i*2,c)^2);
        if force~=0
            % Komponente sile
            Fx=forces(i*2-1,c);
            Fy=forces(i*2,c);
            % koor hvatista
            Xhv=nodes(i,2);
            Yhv=nodes(i,3);
            % koor vrha
            Xvrh=nodes(i,2)+Fx;
            Yvrh=nodes(i,3)-Fy;
            % koor korijena strelice
            Xst=Xvrh-lfac*sign(Fx);
            Yst=Yvrh+lfac*sign(Fy);
            % koor lijevog i desnog ruba strelice
            Xl=Xst+wfac*sign(Fy);
            Yl=Yst+wfac*sign(Fx);
            Xd=Xst-wfac*sign(Fy);
            Yd=Yst-wfac*sign(Fx);

            % Obicna strelica
            if ArrowType=='V'
            plot([Xhv,Xvrh],[Yhv,
Yvrh],boja,'linewidth',deb)
            plot([Xvrh,Xl],[Yvrh,Yl],boja,'linewidth',deb)
            plot([Xvrh,Xd],[Yvrh,Yd],boja,'linewidth',deb)
            end
            % Puna strelica
            if ArrowType=='D'
                plot([Xhv,Xst],[Yhv,
Yst],boja,'linewidth',deb)

plot([Xvrh,Xl,Xst,Xd,Xvrh],[Yvrh,Yl,Yst,Yd,Yvrh],boja,'line
width',deb)
            end
        end
    end
end
% osi moraju biti proporcionalne
axis('equal')
figure(gcf)
```