SVEUČILIŠTE U ZAGREBU
GRAĐEVINSKI FAKULTET

ZAVRŠNI RAD

Inačice Crossova postupka

Ivan Manović
Mentor: *prof. dr. sc. Krešimir Fresl*

Zagreb, rujan 2020.

SADRŽAJ

1. Uvod **1**

2. Crossov postupak **2**

2.1. Opis, definicije i izvodi 2

2.2. Primjer 6

2.3. Jacobijev iteracijski postupak 8

2.3.1. Crossov postupak kao *modificirani Jacobijev postupak* 10

2.4. Gauss-Seidelov iteracijski postupak 11

2.4.1. Crossov postupak kao *modificirani Gauss-Seidelov postupak* . 12

3. Načini obilaženja čvorova **14**

4. Primjeri i analiza broja računskih koraka **19**

5. Zaključak **24**

Literatura **25**

1. Uvod

Kroz povijest, gradnja konstrukcija za razne namjene je bila neizbježna. U početku su to bile jednostavne građevine koje su se gradile intuitivno i iskustveno po pitanju stabilnosti i nosivosti. S vremenom su građevine postajale sve složenije pa je nastala potreba za osnovnim proračunima. U bližoj povijesti uvedena su određena pravila, zakonitosti i terminologija kako bi poznavanje i pristup rješavanju problema bio dosljedniji, općenitiji i univerzalniji. Razvijane su mnoge metode proračuna konstrukcija u težnji za pojednostavljenjem, održanjem dosljednosti i točnosti istih. Kao što je već rečeno, konstrukcije su postale poprilično složene što povlači i složene postupke izračuna velikog broja nepoznanica, uz određene ulazne parametre i određena ograničenja.

Kako bi se izbjeglo iscrpno rješavanje sustava jednadžbi u želji izravnog otkrivanja vrijednosti nepoznanica, mudrim odabirom suvišnih nepoznanica na osnovi pojednostavnjujućih pretpostavaka, iskorištenjem uvjeta simetrije i rubnih uvjeta možemo umanjiti broj nepoznanica do prihvatljive razine. No i dalje, uz određene uvjete, može se olakšati posao određivanja nepoznanica postupnom aproksimacijom. Postupci proračuna koji se temelje na ideji postupne aproksimacije rješenja nazivaju se *iteracijski* odnosno *relaksacijski postupci*.

Ovdje je razložen jedan od relaksacijskih postupaka – *Crossov postupak*. Objašnjen je sâm postupak i povezan je s *Jacobijevim iteracijskim postupkom* i *Gauss-Seidelovim iteracijskim postupkom* rješavanja sustava jednadžbi. Nadalje, uspoređene su inačice Crossova postupka odnosno mogući načini obilaženja i uravnotežavanja čvorova. Kako se čvorovi ne bi obilazili ručno i postupci bili podložni slučajnim greškama, za potrebe ovog rada upotrijebljen je programski alat *SageMath*. U njemu je implementirana pojedina inačica te je ispravnost samog koda utvrđena usporedbom dobivenih rezultata s onima iz programa *RFEM*. U konačnici napravljena je analiza i usporedba inačica provedenih kroz nekoliko primjera.

2. Crossov postupak

2.1. Opis, definicije i izvodi

Crossov postupak ili *metoda raspodjele momenata* je iteracijski odnosno relaksacijski postupak za rješavanje statičkih sustava. Dakle, umjesto da se izravnim putem (npr. rješavanjem jednadžbi *metode pomaka* Gaussovim eliminacijskim postupkom) dolazi do ravnotežnog stanja sustava, u ovom slučaju se postupnim uravnotežavanjem čvorova postiže ravnoteža. Ovim postupkom se iteracijski približava rješenju zadovoljavajuće točnosti. Budući da postupak brzo konvergira, već se nakon nekoliko koraka iteracije postiže prihvatljivo rješenje.

U proračun se ulazi uz nekoliko pretpostavaka i ograničenja. Pod djelovanjem vanjskog opterećenja čvorovi sustava će se zaokrenuti za neke kutove $\varphi_\Diamond$ i pomaknuti za veličinu u, koja je za sve čvorove grede jednaka. U Crossovu postupku će se eventualne pomake u spriječiti dodatnom vezom (za tlo), a čvorovi će se smatrati nepomičnima i u ekvivalentom sustavu će im se spriječiti zaokretanje. Na taj način dobiva se nepomični osnovni sustav.

Vanjsko opterećenje preuzima se momentima upetosti $\overline{M}_{\Diamond,\Diamond}$ na krajevima štapova. Osim zadanog djelovanja, momente na krajevima štapova uzrokuju i zaokreti čvorova, pa je ukupna vrijednost momenta na kraju i štapa $\{i, j\}$ jednaka

$$M_{i,j} = m_{i,j}(\varphi_i, \varphi_j) + \overline{M}_{i,j} = 4k_{i,j}\varphi_i + 2k_{i,j}\varphi_j + \overline{M}_{i,j}. \qquad (2.1)$$

U samome čvoru i djeluju momenti jednakog iznosa, ali suprotnoga smjera kao što su na krajevima priključenih štapova $\{i, j_i\}$ i eventualni koncentrirani vanjski moment $\overline{M}_i$. Ravnotežu čvora nalaže uvjet

$$\sum_{j_i}(-M_{i,j_i}) + \overline{M}_i = 0.$$

Ako s M_i označimo zbroj svih poznatih nam momenata

$$M_i = \sum_{j_i}(-\overline{M}_{i,j_i}) + \overline{M}_i,$$

jednadžba ravnoteže čvora poprima oblik

$$-\left(\sum_{j_i} 4k_{i,j_i}\right)\varphi_i - \sum_{j_i} 2k_{i,j_i}\varphi_{j_i} + M_i = 0. \tag{2.2}$$

Kutovi $\varphi_\Diamond$ moraju zadovoljavati jednadžbu (2.2) za svaki čvor konstrukcije. Umjesto izravnog izračunavanja pravih vrijednosti $\varphi_\Diamond$, može im se postupno približavati. Pretpostavkom kutova zaokreta $\varphi_\Diamond^{(n_\Diamond)}$ susjednih čvorova narušava se ravnoteža promatranog čvora za $M_\Diamond^{(n_\Diamond)}$. Općenito će vrijediti

$$-\left(\sum_{j_i} 4k_{i,j_i}\right)\varphi_i^{(n_i)} - \sum_{j_i} 2k_{i,j_i}\varphi_{j_i}^{(n_{j_i})} + M_i = M_i^{(n_i)} \tag{2.3}$$

gdje je $M_i^{(n_i)}$ neuravnoteženi ili rezidualni moment. Prave vrijednosti koje zadovoljavaju jednadžbu (2.2) razlikuju se od kutova $\varphi_\Diamond^{(n_\Diamond)}$ za priraste $\Delta\varphi_\Diamond$:

$$-\left(\sum_{j_i} 4k_{i,j_i}\right)\left(\varphi_i^{(n_i)} + \Delta\varphi_i\right) - \sum_{j_i} 2k_{i,j_i}\left(\varphi_{j_i}^{(n_{j_i})} + \Delta\varphi_{j_i}\right) + M_i = 0.$$

Oduzme li se od te jednadžbe jednadžba (2.3), dobiva se rezidualna jednadžba

$$-\left(\sum_{j_i} 4k_{i,j_i}\right)\Delta\varphi_i - \sum_{j_i} 2k_{i,j_i}\Delta\varphi_{j_i}^{(n_{j_i})} + M_i^{(n_i)} = 0. \tag{2.4}$$

Pri ponovnom uravnotežavanju opet se ostali čvorovi pričvrste, a promatrani čvor dodatno zakrene, pa je prirast kuta potreban za uravnoteženje

$$\Delta\varphi_i^{(n_i+1)} = \frac{M_i^{(n_i)}}{\sum_{j_i} 4k_{i,j_i}}. \tag{2.5}$$

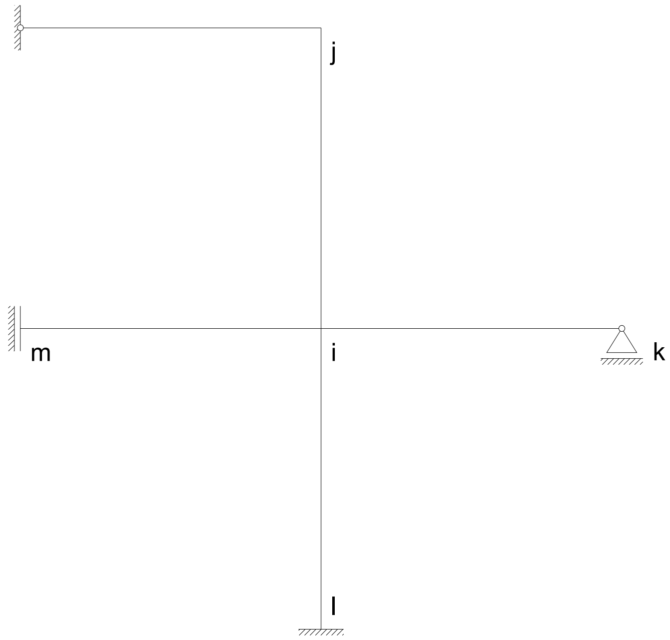
Izraz (2.5) se može upotrijebiti u slučaju kada su štapovi $\{i, j_i\}$ obostrano upeti. Ako nisu svi štapovi obostrano upeti, onda je izraz malo modificiran. Za primjer prikazan na slici 2.1 će vrijediti malo drugačija formula uvjetovana rubnim uvjetima na krajevima j_i štapova $\{i, j_i\}$:

$$\Delta\varphi_i^{(n_i+1)} = \frac{M_i^{(n_i)}}{4k_{i,j} + 4k_{i,l} + 3k_{i,k} + k_{i,m}}.$$

Iskoriste li se jednadžbe (2.1) i (2.5), mogu se dobiti sljedeća dva izraza:

$$\Delta M_{i,j_i}^{(n_i+1)} = 4k_{i,j_i}\Delta\varphi_i^{(n_i+1)} = \frac{4k_{i,j_i}}{\sum_{s_i} 4k_{i,s_i}} M_i^{(n_i)}, \tag{2.6}$$

$$\Delta M_{j_i,i}^{(n_i+1)} = 2k_{j_i,i}\Delta\varphi_i^{(n_i+1)} = \frac{1}{2}\Delta M_{i,j_i}^{(n_i+1)}. \tag{2.7}$$



Slika 2.1

Iz prethodnih jednakosti dâ se zaključiti kako promjenu kuta zaokreta prati promjena momenta, pa se brojčano izračunavanje kutova $\varphi_\Diamond$ može izostaviti. Momenti se mogu odmah računati zbog prolaznosti značenja vrijednosti kutova zaokreta. Prethodno navedeno razlikuje Crossov postupak od *Čališavljevog*.

Neka je koeficijent krutosti čvora i definiran sa

$$k_i = \sum_{j_i} 4k_{i,j_i},$$

a razdjelni koeficijent u čvoru i za štap $\{i, j_i\}$ definiran sa

$$\mu_{i,j_i} = \frac{4k_{i,j_i}}{k_i}.$$

Očito vrijedi

$$\sum_{j_i} \mu_{i,j_i} = 1.$$

Uvrštavanjem izraza za μ_{i,j_i} u jednadžbu (2.6) dobiva se

$$\Delta M_{i,j_i}^{(n_i+1)} = \mu_{i,j_i} M_i^{(n_i)}$$

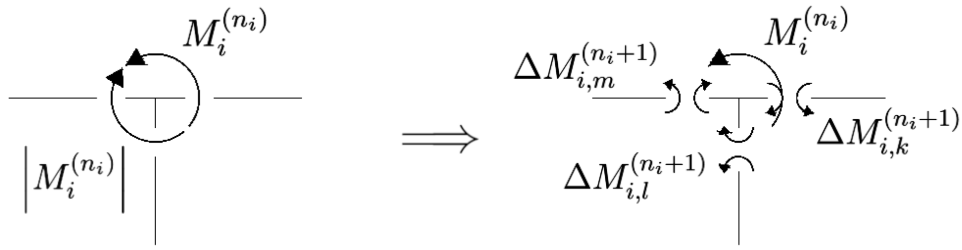
iz čega slijedi

$$\sum_{j_i} \Delta M_{i,j_i}^{(n_i+1)} = M_i^{(n_i)}$$

odnosno

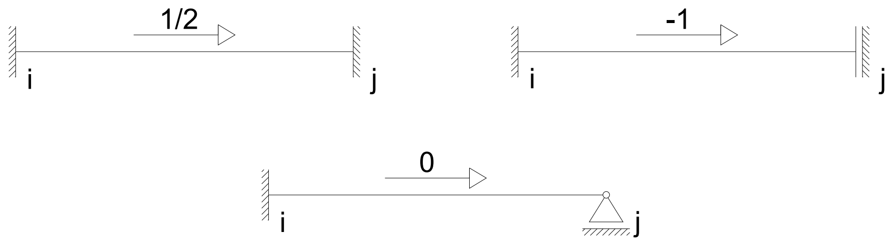
$$-\sum_{j_i} \Delta M_{i,j_i}^{(n_i+1)} + M_i^{(n_i)} = 0. \tag{2.8}$$

Izraz (2.8) pokazuje kako je moguće uravnotežiti čvor i dodavajući moment jednakog intenziteta, ali suprotnog smisla vrtnje rezidualnog momenta $M_i^{(n_i)}$. Dodani moment se, nadalje, raspodjeljuje na priključene štapove u omjerima njihovih krutosti (slika 2.2). Navedeno upućuje na dodatni naziv Crossova postupka – *postupak raspodjele momenata*.



Slika 2.2

U jednadžbi (2.7) se vidi kako uravnoteženje čvora i narušava ravnotežu susjednih čvorova. Točnije rečeno, polovina vrijednosti momenta na kraju i štapa $\{i, j_i\}$ se prenosi na kraj j – *prijenosni koeficijent* jednak je 1/2. Sve prethodno vrijedi za slučaj obostrano upetih štapova. Ako je štap jednostrano upet ili mu je na jednom kraju klizno upeti ležaj, prijenosni koeficijenti su 0 i -1 redom (slika 2.3).



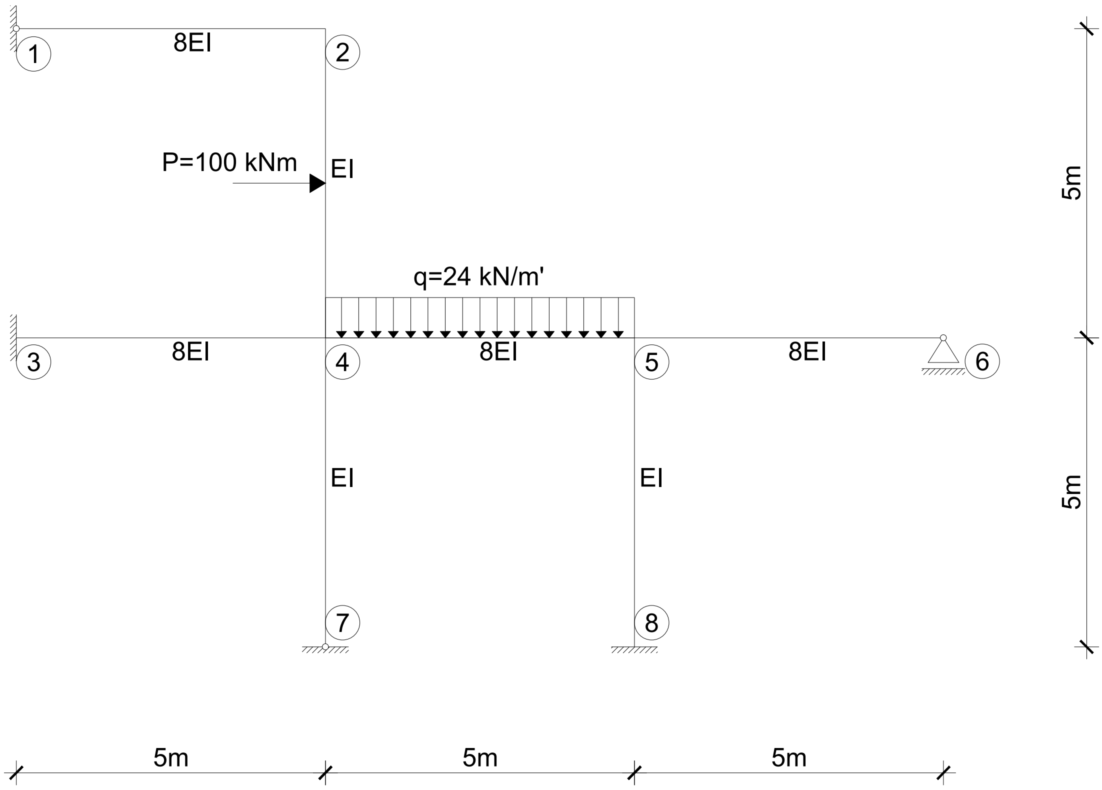
Slika 2.3

Crossov je postupak relaksacijski, jer se pri svakom uravnoteženju izračunavaju prirasti vrijednosti momenata. Konačne vrijednosti momenata se dobiju tako da se na svakom kraju štapa zbroje vrijednosti svih momenata – momenata upetosti, raspodijeljenih i prenesenih momenata koji su se na njemu pojavili.

U nastavku rada bit će navedeni i opisani mogući redoslijedi obilazaka čvorova, prednosti i nedostaci pojedine inačice te usporedba brzine konvergencije.

2.2. Primjer

U nastavku je riješen primjer sa slike 2.4 Crossovim postupkom.



Slika 2.4

Prvo se određuju krutosti štapova

$$k_{1,2} = k_{3,4} = k_{4,5} = k_{5,6} = \tfrac{8EI}{5},$$
$$k_{2,4} = k_{4,7} = k_{5,8} = \tfrac{EI}{5},$$

i krutosti čvorova

$$k_2 = 3k_{1,2} + 4k_{2,4} = \tfrac{3\cdot 8EI}{5} + \tfrac{4\cdot EI}{5} = \tfrac{28EI}{5},$$
$$k_4 = 4k_{2,4} + 4k_{3,4} + 4k_{4,5} + 3k_{4,7} = \tfrac{4\cdot EI}{5} + \tfrac{4\cdot 8EI}{5} + \tfrac{4\cdot 8EI}{5} + \tfrac{3\cdot EI}{5} = \tfrac{71EI}{5},$$
$$k_5 = 4k_{4,5} + 4k_{5,8} + 3k_{5,6} = \tfrac{4\cdot 8EI}{5} + \tfrac{4\cdot EI}{5} + \tfrac{3\cdot 8EI}{5} = \tfrac{60EI}{5}.$$

Nadalje, dobivaju se razdjelni koeficijenti:

$$\mu_{2,1} = \tfrac{3k_{1,2}}{k_2} = \tfrac{6}{7}, \qquad \mu_{4,2} = \tfrac{4k_{2,4}}{k_4} = \tfrac{4}{71}, \qquad \mu_{5,4} = \tfrac{4k_{4,5}}{k_5} = \tfrac{8}{15},$$
$$\mu_{2,4} = \tfrac{4k_{2,4}}{k_2} = \tfrac{1}{7}, \qquad \mu_{4,3} = \tfrac{4k_{3,4}}{k_4} = \tfrac{32}{71}, \qquad \mu_{5,8} = \tfrac{4k_{5,8}}{k_5} = \tfrac{1}{15},$$
$$\mu_{4,5} = \tfrac{4k_{4,5}}{k_4} = \tfrac{32}{71}, \qquad \mu_{5,6} = \tfrac{3k_{5,6}}{k_5} = \tfrac{6}{15}.$$
$$\mu_{4,7} = \tfrac{3k_{4,7}}{k_4} = \tfrac{3}{71},$$

Na osnovi zadanog vanjskog opterećenja $P = 100$ kN i $q = 24$ kN/m' izračunavaju se momenti upetosti

$$\overline{M}_{4,2} = -\overline{M}_{2,4} = \frac{P \cdot l_{2,4}}{8} = \frac{100 \cdot 5}{8} = 62,5 \text{ kNm},$$
$$\overline{M}_{4,5} = -\overline{M}_{5,4} = \frac{q \cdot l_{4,5}^2}{12} = \frac{24 \cdot 5^2}{12} = 50 \text{ kNm}.$$

Nakon što se odrede početni rezidualni momenti

$$M_2^{(0)} = -62,5 \text{ kNm},$$
$$M_4^{(0)} = 112,5 \text{ kNm},$$
$$M_5^{(0)} = -50 \text{ kNm},$$

za potrebe ovog primjera odabrat će se čvor s najvećim rezidualnim momentom. Suprotna vrijednost rezidualnog momenta se podijeli na uravnotežujuće momente u omjerima razdjelnih koeficijenata:

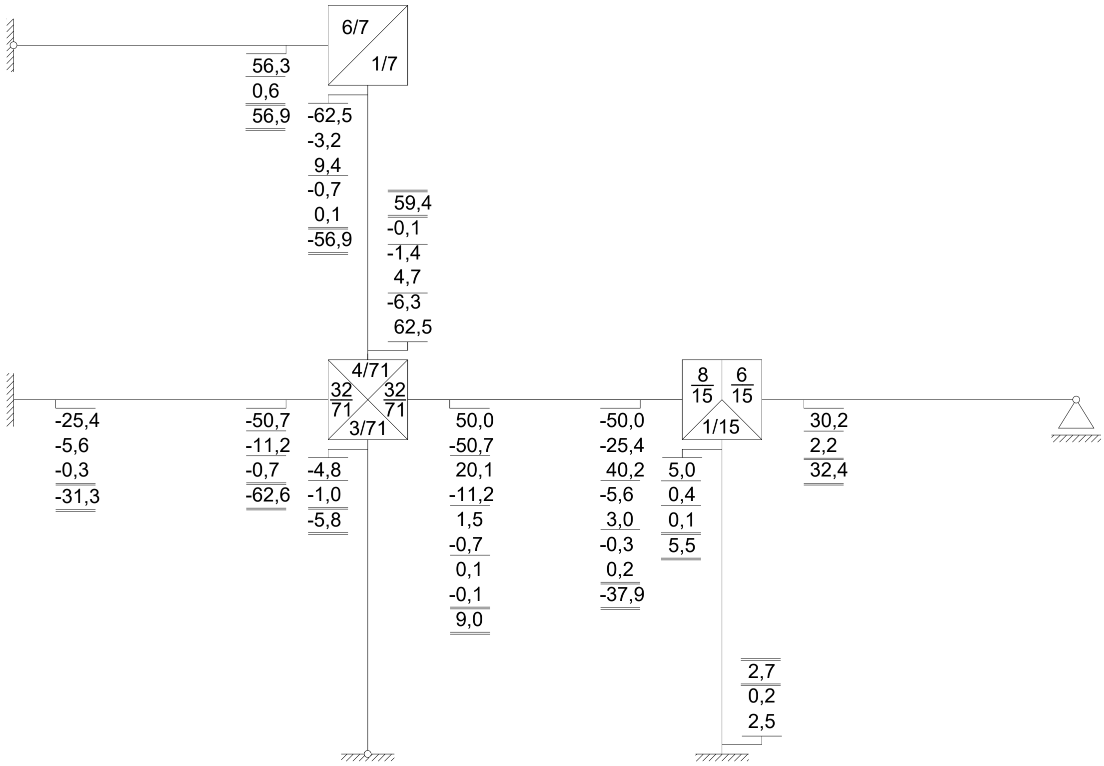
$$\Delta M_{4,2}^{(1)} = \mu_{4,2} \cdot (-M_4^{(0)}) = \tfrac{4}{71} \cdot (-112,5) = -6,3 \text{ kNm},$$
$$\Delta M_{4,3}^{(1)} = \mu_{4,3} \cdot (-M_4^{(0)}) = \tfrac{32}{71} \cdot (-112,5) = -50,7 \text{ kNm},$$
$$\Delta M_{4,5}^{(1)} = \mu_{4,5} \cdot (-M_4^{(0)}) = \tfrac{32}{71} \cdot (-112,5) = -50,7 \text{ kNm},$$
$$\Delta M_{4,7}^{(1)} = \mu_{4,7} \cdot (-M_4^{(0)}) = \tfrac{3}{71} \cdot (-112,5) = -4,8 \text{ kNm}.$$

Prepoznaje se da je $\sum_{j_4} \Delta M_{4,j_4}^{(1)} = -M_4^{(0)} = -112,5$ kNm. Zadovoljen je izraz (2.8) što znači da je čvor u ravnoteži. Sljedeći korak je prijenos dijelova momenata na susjedne čvorove:

$$M_2^{(1)} = M_2^{(0)} + \tfrac{1}{2}\Delta M_{4,2}^{(1)} = -62,5 + \tfrac{1}{2} \cdot (-6,3) = -65,7 \text{ kNm},$$
$$M_3^{(1)} = M_3^{(0)} + \tfrac{1}{2}\Delta M_{4,3}^{(1)} = 0 + \tfrac{1}{2} \cdot (-50,7) = -50,7 \text{ kNm},$$
$$M_5^{(1)} = M_5^{(0)} + \tfrac{1}{2}\Delta M_{4,5}^{(1)} = -50 + \tfrac{1}{2} \cdot (-50,7) = -75,4 \text{ kNm}.$$

Dakle, dobiveni su novi rezidualni momenti nakon prvog koraka iteracije. Drugim korakom iteracije se opet uravnotežuje čvor s najvećim rezidualnim momentom. Nastavak iteriranja je prikazan na slici 2.5. Nakon što se postigne zadovoljavajuća točnost (rezidualni momenti $M_i \to 0$), na svakome kraju štapa se zbroje svi ikad pojavljivani momenti tokom izvođenja postupka. Dobiveni zbrojevi predstavljaju konačne momente.

Zadani primjer je nepomični sustav. Stoga se odmah dobivaju konačni momenti nakon prvog uravnotežavanja čvorova. U slučaju da su u sustavu dozvoljeni translacijski pomaci, postupak se provodi u dva dijela. U prvom dijelu se rješava nepomični sustav koji nastaje sprečavanjem pomaka dodavanjem pridržanja. Dobivene momente od stvarnih razlikuje utjecaj pridržanja. U pridržanjima se javljaju reakcije koje se mogu izračunati iz ravnoteže horizontalnih sila. U nastavku se sustav opterećuje reak-



Slika 2.5

cijama suprotnog smjera na mjestima pridržanja. Na taj se način poništavaju reakcije pridržanja. Do konačnih momenata se dolazi *proširenjem Crossova postupka* ili pak nekim drugim relaksacijskim postupkom.

2.3. Jacobijev iteracijski postupak

Jacobijev iteracijski postupak je iteracijska metoda rješavanja sustava linearnih jednadžbi oblika:

$$a_{11}x_1 + a_{12}x_2 + \cdots + a_{1n}x_n = b_1$$
$$a_{21}x_1 + a_{22}x_2 + \cdots + a_{2n}x_n = b_2$$
$$\vdots$$
$$a_{n1}x_1 + a_{n2}x_2 + \cdots + a_{nn}x_n = b_n.$$

Isti sustav se može zadati i u matričnom obliku $\mathbf{Ax} = \mathbf{b}$ gdje je

$$\mathbf{A} = \begin{bmatrix} a_{11} & a_{12} & \cdots & a_{1n} \\ a_{21} & a_{22} & \cdots & a_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ a_{n1} & a_{n2} & \cdots & a_{nn} \end{bmatrix}, \quad \mathbf{x} = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix} \quad \text{i} \quad \mathbf{b} = \begin{bmatrix} b_1 \\ b_2 \\ \vdots \\ b_n \end{bmatrix}.$$

Neka su elementi na glavnoj dijagonali matrice $\mathbf{A}$ različiti od nule. Rastavimo $\mathbf{A}$ tako da je $\mathbf{A} = \mathbf{L} + \mathbf{D} + \mathbf{R}$ gdje su

$$\mathbf{L} = \begin{bmatrix} 0 & 0 & \cdots & 0 \\ a_{21} & 0 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ a_{n1} & a_{n2} & \cdots & 0 \end{bmatrix}, \quad \mathbf{D} = \begin{bmatrix} a_{11} & 0 & \cdots & 0 \\ 0 & a_{22} & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & a_{nn} \end{bmatrix}$$

$$\text{i} \;\; \mathbf{R} = \begin{bmatrix} 0 & a_{12} & \cdots & a_{1n} \\ 0 & 0 & \cdots & a_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & 0 \end{bmatrix}.$$

$\mathbf{L}$ je donja, a $\mathbf{R}$ gornja trokutasta matrica s nulama na glavnoj dijagonali dok je $\mathbf{D}$ dijagonalna matrica. Sada se izraz $\mathbf{Ax} = \mathbf{b}$ može prikazati kao

$$\mathbf{Dx^{(k+1)}} = -(\mathbf{L} + \mathbf{R})\mathbf{x^{(k)}} + \mathbf{b}, \qquad k = 0, 1, 2, ...$$

Budući da su elementi na glavnoj dijagonali matrice $\mathbf{D}$ različiti od nule, postoji inverzna matrica $\mathbf{D^{-1}}$. Pomnoži li se prethodni izraz s lijeva s $\mathbf{D^{-1}}$, dobiva se da je

$$\mathbf{x^{(k+1)}} = -\mathbf{D^{-1}}(\mathbf{L} + \mathbf{R})\mathbf{x^{(k)}} + \mathbf{D^{-1}b}. \tag{2.9}$$

Izraz (2.9) je ekvivalentan sustavu

$$\begin{aligned} x_1^{(k+1)} &= \alpha_{12}x_2^{(k)} + \alpha_{13}x_3^{(k)} + \cdots + \alpha_{1n}x_n^{(k)} + \beta_1 \\ x_2^{(k+1)} &= \alpha_{21}x_1^{(k)} + \alpha_{23}x_3^{(k)} + \cdots + \alpha_{2n}x_n^{(k)} + \beta_2 \\ &\vdots \\ x_n^{(k+1)} &= \alpha_{n1}x_1^{(k)} + \alpha_{n2}x_2^{(k)} + \cdots + \alpha_{nn-1}x_{n-1}^{(k)} + \beta_n, \end{aligned}$$

gdje je $\alpha_{ij} = -\frac{a_{ij}}{a_{ii}}$ i $\beta_i = \frac{b_i}{a_{ii}}$. Iz prethodnog vrijedi da je

$$x_i^{(k+1)} = \sum_{j \neq i}^{n} \alpha_{ij}x_j^{(k)} + \beta_i = \frac{1}{a_{ii}}\left(b_i - \sum_{j \neq i}^{n} a_{ij}x_j^{(k)}\right), \qquad i = 1, 2, ..., n. \tag{2.10}$$

Jacobijeva metoda podrazumijeva rješavanje prve jednadžbe po x_1, druge jednadžbe po x_2 i tako do n-te jednadžbe gdje se računa x_n. Započinje se tako da se aproksimira rješenje sustava $\mathbf{x^{(0)}} = \left[x_1^{(0)}, x_2^{(0)}, ..., x_n^{(0)}\right]^T$. Iz početne aproksimacije prvim korakom iteracije se dobiva prvo približno rješenje $\mathbf{x^{(1)}} = \left[x_1^{(1)}, x_2^{(1)}, ..., x_n^{(1)}\right]^T$. Iteriranje se nastavlja sve dok se ne postigne zadovoljavajuća vrijednost $\mathbf{x}$ ili određeni broj koraka.

Iz izraza (2.10) se vidi da se jednadžbe mogu rješavati bilo kojim redoslijedom, jer k-ti korak iteracijskog postupka ovisi samo o prethodnim koracima, a ne o rezultatima iz istog koraka iteracije. Niz $\mathbf{x^{(0)}}, \mathbf{x^{(1)}}, ..., \mathbf{x^{(k)}}$ konvergira ka $\mathbf{x}$, ako i samo ako je $\lim_{k\to\infty} x_i^k = x_i, \forall i = 1, 2, ..., n$. Konkretno, Jacobijev iteracijski postupak će konvergirati ako je promatrana matrica dijagonalno dominantna (ponekad može konvergirati i izvan uvjeta). Uvjet konvergencije je $|a_{ii}| \geq \sum_{j\neq i} |a_{ij}|, \forall i = 1, 2, ..., n$, uz '>' za barem jedan i. Konvergencija ne ovisi o odabiru početne aproksimacije $\mathbf{x^{(0)}}$.

2.3.1. Crossov postupak kao *modificirani Jacobijev postupak*

Riješiti neki statički sustav je zapravo rješavanje jednadžbe $\mathbf{Ku} = \mathbf{m}$ gdje je $\mathbf{K}$ matrica krutosti sustava, $\mathbf{u}$ je vektor stupac kutova zaokreta, a $\mathbf{m}$ je vektor stupac momenata u čvoru:

$$\mathbf{K} = \begin{bmatrix} k_{11} & k_{12} & \cdots & k_{1n} \\ k_{21} & k_{22} & \cdots & k_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ k_{n1} & k_{n2} & \cdots & k_{nn} \end{bmatrix}, \quad \mathbf{b} = \begin{bmatrix} \varphi_1 \\ \varphi_2 \\ \vdots \\ \varphi_n \end{bmatrix} \quad \text{i} \quad \mathbf{m} = \begin{bmatrix} M_1 \\ M_2 \\ \vdots \\ M_n \end{bmatrix}.$$

Sustav jednadžbi neka bude dosljedan onome u Jacobijevu postupku:

$$\begin{aligned} k_{11}\varphi_1^{(k+1)} &= -\left(k_{12}\varphi_2^{(k)} + k_{13}\varphi_3^{(k)} + \cdots + k_{1n}\varphi_n^{(k)}\right) + M_1 \\ k_{22}\varphi_2^{(k+1)} &= -\left(k_{21}\varphi_1^{(k)} + k_{23}\varphi_3^{(k)} + \cdots + k_{2n}\varphi_n^{(k)}\right) + M_2 \\ &\vdots \\ k_{nn}\varphi_n^{(k+1)} &= -\left(k_{n1}\varphi_1^{(k)} + k_{n2}\varphi_2^{(k)} + \cdots + k_{nn-1}\varphi_{n-1}^{(k)}\right) + M_n, \end{aligned}$$

odnosno

$$k_{ii}\varphi_i^{(k+1)} = -\sum_{j\neq i}^{n} k_{ij}\varphi_j^{(k)} + M_i. \tag{2.11}$$

Pošto je $k_{i,i} = \sum_j 4k_{i,j}$ i $M_{i,j} = 2k_{i,j}\varphi_j$, gornji izraz može izgledati kao

$$M_i^{(k+1)} = M_i - \sum_{j\neq i}^{n} \frac{M_{i,j}^{(k)}}{2}.$$

Iz prethodne jednadžbe vidimo kako novi moment u čvoru nastaje promjenom postojećeg momenta za iznos sume prenesenih momenata sa susjednih čvorova. Dâ se uočiti kako su svi preneseni momenti dobiveni u prethodnoj iteraciji, što je karakteristično za Jacobijev iteracijski postupak.

Kad bi se izraz (2.11) oduzeo od jednadžbe $k_{ii}\varphi_i^{(k+2)} = -\sum_{j\neq i}^{n} k_{ij}\varphi_j^{(k+1)} + M_i$ (sljedećega koraka iteracije), nastao bi izraz

$$k_{ii}\Delta\varphi_i^{(k+1)} = -\sum_{j\neq i}^{n} k_{ij}\Delta\varphi_j^{(k)},$$

ako je $\Delta\varphi_i^{(k)} = \varphi_i^{(k+1)} - \varphi_i^{(k)}$. Nadalje, kad bi se dobiveni izraz pomnožio s $4k_{ij}$ i podijelio s k_{ii}, dobilo bi se

$$4k_{ij}\Delta\varphi_i^{(k+1)} = \frac{-4k_{ij}}{k_{ii}}\sum_{j\neq i}^{n} k_{ij\Delta}\varphi_j^{(k)}.$$

Uzme li se u obzir da je $k_{i,i} = \sum_j 4k_{i,j}$, $\Delta M_{i,j} = 2k_{i,j}\Delta\varphi_j$ i $\Delta M_{i,j} = 4k_{ij}\Delta\varphi_i$, konačno se dobiva

$$\Delta M_{i,j}^{(k+1)} = \frac{-k_{ij}}{\sum_j k_{i,j}}\sum_{j\neq i}^{n}\frac{\Delta M_{i,j}^{(k)}}{2}. \tag{2.12}$$

Jednadžba (2.12) je opći oblik Crossova postupka za istodobno uravnotežavanje svih čvorova sustava. Ista jednadžba je ekvivalentna jednadžbi (2.10). Prema tome, ako se u Crossovu postupku istovremeno uravnotežavaju svi čvorovi iz iteracije u iteraciju, može ga se smatrati modifikacijom Jacobijeva postupka.

2.4. Gauss-Seidelov iteracijski postupak

Gauss-Seidelova metoda je jako slična Jacobijevoj metodi. I u ovom slučaju riječ je o iteracijskom postupku gdje se postupnim aproksimacijama dolazi do rješenja.

Opet promatramo sustav jednadžbi matrično zapisan $\mathbf{Ax} = \mathbf{b}$. Neka vrijede sve iste pretpostavke kao i na početku odjeljka *2.3. Jacobijev iteracijski postupak*. Vrijedi:

$$(\mathbf{L}+\mathbf{D})\mathbf{x}^{(\mathbf{k+1})} = -\mathbf{R}\mathbf{x}^{(\mathbf{k})} + \mathbf{b}, \qquad \mathbf{k} = \mathbf{0}, \mathbf{1}, \mathbf{2}, ...$$

Intuitivnije je navedeni izraz odmah množiti s lijeva inverznom matricom $(\mathbf{L}+\mathbf{D})^{-\mathbf{1}}$, ali ovaj put je bolje pomnožiti s lijeva inverznom matricom $\mathbf{D}^{-\mathbf{1}}$, jer ju je lakše pronaći. Dobiva se

$$(\mathbf{D}^{-\mathbf{1}}\mathbf{L}+\mathbf{I})\mathbf{x}^{(\mathbf{k+1})} = -\mathbf{D}^{-\mathbf{1}}\mathbf{R}\mathbf{x}^{(\mathbf{k})} + \mathbf{D}^{-\mathbf{1}}\mathbf{b},$$

iz čega slijedi

$$\mathbf{x}^{(\mathbf{k+1})} = -\mathbf{D}^{-\mathbf{1}}\mathbf{L}\mathbf{x}^{(\mathbf{k+1})} - \mathbf{D}^{-\mathbf{1}}\mathbf{R}\mathbf{x}^{(\mathbf{k})} + \mathbf{D}^{-\mathbf{1}}\mathbf{b}. \tag{2.13}$$

Kada bi se jednadžba (2.13) raspisala u obliku sustava jednadžbi to bi izgledalo ovako:

$$\begin{aligned}
x_1^{(k+1)} &= \alpha_{12}x_2^{(k)} + \alpha_{13}x_3^{(k)} + \cdots + \alpha_{1n}x_n^{(k)} + \beta_1 \\
x_2^{(k+1)} &= \alpha_{21}x_1^{(k+1)} + \alpha_{23}x_3^{(k)} + \cdots + \alpha_{2n}x_n^{(k)} + \beta_2 \\
&\vdots \\
x_n^{(k+1)} &= \alpha_{n1}x_1^{(k+1)} + \alpha_{n2}x_2^{(k+1)} + \cdots + \alpha_{nn-1}x_{n-1}^{(k+1)} + \beta_n.
\end{aligned}$$

Prethodni sustav daje izraz

$$x_i^{(k+1)} = \sum_{j=1}^{i-1} \alpha_{ij}x_j^{(k+1)} + \sum_{j=i+1}^{n} \alpha_{ij}x_j^{(k)} + \beta_i.$$

Ako se uzme u obzir da je $\alpha_{ij} = -\frac{a_{ij}}{a_{ii}}$ i $\beta_i = \frac{b_i}{a_{ii}}$, onda gornji izraz poprima oblik

$$x_i^{(k+1)} = \frac{1}{a_{ii}} \left(b_i - \sum_{j=1}^{i-1} a_{ij}x_j^{(k+1)} - \sum_{j=i+1}^{n} a_{ij}x_j^{(k)} \right), \qquad i = 1, 2, ..., n. \tag{2.14}$$

Iz jednadžbe (2.13) raspisane u obliku sustava jednadžbi se može uvidjeti što nalaže Gauss-Seidelov postupak. Kao i kod Jacobijevog postupka u prvoj jednadžbi se traži x_1, u drugoj x_2 i tako dalje. Započinje se tako što se u prvu jednadžbu ulazi sa inicijalnim aproksimacijama $x_2^{(0)}, x_3^{(0)}, ..., x_n^{(0)}$. No za razliku od Jacobijevog postupka u kojem se u svaku jednadžbu ulazi s inicijalnom aproksimacijom, u drugu jednadžbu uvrštavamo $x_1^{(1)}, x_3^{(0)}, ..., x_n^{(0)}$ i tako sve do posljednje n-te jednadžbe u koju uvrštavamo $x_1^{(1)}, x_2^{(1)}, ..., x_{n-1}^{(1)}$. Postupak konvergira uz iste uvjete kao i Jacobijev postupak.

Ako iteracijski postupak konvergira, jasno je da će $\mathbf{x}^{(\mathbf{k+1})}$ biti bliže $\mathbf{x}$ nego što će to biti $\mathbf{x}^{(\mathbf{k})}$. Dakle, brže se dolazi do rješenja ako se za izračunavanje $x_{i+1}^{(k+1)}$ uvrštavaju "novije" vrijednosti $x_1^{(k+1)}, x_2^{(k+1)}, ..., x_i^{(k+1)}$. Prema tome može se zaključiti da Gauss-Seidelov postupak brže konvergira od Jacobijevog (postoje iznimke primjera matrica gdje to nije tako ili gdje se konvergencija uopće ne ostvaruje).

2.4.1. Crossov postupak kao *modificirani Gauss-Seidelov postupak*

Kao i u pododjeljku 2.3.1. promatra se izraz $\mathbf{K}\mathbf{u} = \mathbf{m}$ gdje je $\mathbf{K}$ matrica krutosti sustava, $\mathbf{u}$ je vektor stupac kutova zaokreta, a $\mathbf{m}$ je vektor stupac momenata u čvoru:

$$\mathbf{K} = \begin{bmatrix} k_{11} & k_{12} & \cdots & k_{1n} \\ k_{21} & k_{22} & \cdots & k_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ k_{n1} & k_{n2} & \cdots & k_{nn} \end{bmatrix}, \quad \mathbf{b} = \begin{bmatrix} \varphi_1 \\ \varphi_2 \\ \vdots \\ \varphi_n \end{bmatrix} \quad \text{i} \quad \mathbf{m} = \begin{bmatrix} M_1 \\ M_2 \\ \vdots \\ M_n \end{bmatrix}.$$

Neka promatrani sustav jednadžbi bude po uzoru na onaj Gauss-Seidelova postupka:

$$\begin{aligned}
k_{11}\varphi_1^{(k+1)} &= -\left(k_{12}\varphi_2^{(k)} + k_{13}\varphi_3^{(k)} + \cdots + k_{1n}\varphi_n^{(k)}\right) + M_1 \\
k_{22}\varphi_2^{(k+1)} &= -\left(k_{21}\varphi_1^{(k+1)} + k_{23}\varphi_3^{(k)} + \cdots + k_{2n}\varphi_n^{(k)}\right) + M_2 \\
&\vdots \\
k_{nn}\varphi_n^{(k+1)} &= -\left(k_{n1}\varphi_1^{(k+1)} + k_{n2}\varphi_2^{(k+1)} + \cdots + k_{nn-1}\varphi_{n-1}^{(k+1)}\right) + M_n.
\end{aligned}$$

Vrijedi jednakost

$$k_{ii}\varphi_i^{(k+1)} = -\sum_{j=i}^{i-1} k_{ij}\varphi_j^{(k+1)} - \sum_{j=i+1}^{n} k_{ij}\varphi_j^{(k)} + M_i. \tag{2.15}$$

Za $k_{i,i} = \sum_j 4k_{i,j}$ i $M_{i,j} = 2k_{i,j}\varphi_j$, gornji izraz postaje

$$M_i^{(k+1)} = M_i - \sum_{j=i}^{i-1} \frac{M_{i,j}^{(k+1)}}{2} - \sum_{j=i+1}^{n} \frac{M_{i,j}^{(k)}}{2}.$$

U odnosu na modifikaciju Jacobijeve metode, ovdje se pojavio još jedan član u izrazu. Zapravo suma koja se javlja kod modifikacije Jacobijeve metode je u ovom slučaju podijeljena u dvije sume – u jednoj se zbrajaju preneseni momenti prethodnog, a u drugoj trenutnog koraka iteracije.

Iz izraza (2.15), na isti način kao i u modifikaciji Jacobijeve metode, može se dobiti da je

$$k_{ii}\Delta\varphi_i^{(k+1)} = -\sum_{j=1}^{i-1} k_{ij}\Delta\varphi_j^{(k+1)} - \sum_{j=i+1}^{n} k_{ij}\Delta\varphi_j^{(k)},$$

odnosno

$$\Delta M_{i,j}^{(k+1)} = \frac{-k_{ij}}{\sum_j k_{i,j}} \left(\sum_{j=1}^{i-1} \frac{\Delta M_{i,j}^{(k+1)}}{2} + \sum_{j=i+1}^{n} \frac{\Delta M_{i,j}^{(k)}}{2} \right). \tag{2.16}$$

Jednadžba (2.16) prikazuje opći oblik Crossova postupka u kojemu se uravnotežava čvorove s "najnovijim" izračunatim momentima. Dâ se primjetiti kako je jednadžba (2.16) zapravo modificirana jednadžba (2.14) koja predstavlja Gauss-Seidelov iteracijski postupak.

3. Načini obilaženja čvorova

Glavni ulazni parametri u dimenzioniranju konstrukcije su unutarnja naprezanja konstrukcije. Unutarnja naprezanja su posljedica uravnoteženja sustava pod utjecajem vanjskog opterećenja. Dakle, ključni korak za projektiranje stabilne konstrukcije je upravo određivanje unutarnjih naprezanja – momenata u čvorovima. Već je spomenuto kako postoje izravne i neizravne metode proračuna. U ovom radu je razrađena jedna od neizravnih metoda – Crossova iteracijska metoda.

Crossov postupak konvergira. Može se, između ostaloga, uočiti kako redoslijed obilaženja čvorova odnosno redoslijed rješavanja jednadžbi ravnoteže čvorova ne utječe na konvergenciju. Drugim riječima, kad-tad će se doći do rješenja. Stoga se postavlja pitanje *koji su mogući redoslijedi?*

U prošlom poglavlju razjašnjena su dva osnovna principa iteriranja – Jacobijev i Gauss-Seidelov. Kod Jacobijevog postupka, samim time i kod Crossova postupka kao njegove modifikacije, uočava se da se rješavanje jednadžbi odnosno uravnotežavanje čvorova može obavljati istodobno. Iz toga je jednostavno zaključiti da se neovisno o redoslijedu obilaženja čvorova dolazi do tražene točnosti rješenja s uvijek istim brojem koraka iteracije. Kako se parametri iz jednadžbe u jednadžbu "osvježavaju", u Gauss-Seidelovoj verziji se redoslijedi obilaska čvorova razlikuju po pitanju broja potrebnih koraka iteracije. Stoga, u nastavku, ima smisla "baciti u isti koš" Crossov postupak kao modificirani Jacobijev postupak i inačice Crossova postupka na osnovi Gauss-Seidelova postupka. U nastavku su navedene neke od mogućih inačica Crossova postupka koje su za potrebu analize broja koraka implementirane u programskom alatu *SageMath*. Pored samog opisa za svaku inačicu postupka priložen je i odgovarajući pseudokod.

Prije svega, prvo treba definirati neke varijable i pomoćne funkcije koje će se, nadalje, upotrebljavati u pseudokodovima inačica:

1. *nodesForVisit* – lista čvorova sustava koji se relaksiraju;
2. *residualMoments* – mapa pripadajućih vrijednosti rezidualnih momenata za svaki čvor sustava;
3. CALCULATE_RESIDUALS – pomoćna funkcija koja rekalkulira rezidualne momente u čvorovima;
4. RELAX_STEP – pomoćna funkcija koja za zadani čvor obavlja postupak relaksacije (uravnotežuje ga);
5. MAX – pomoćna funkcija koja vraća čvor sa najvećim rezidualnim momentom;
6. ERROR_FUNCTION – proizvoljna matematička funkcija pogreške kojom se definira udaljenost od točnog rješenja; vrijednost može poslužiti kao uvjet za zaustavljanje postupka;
7. SHUFFLE – funkcija koja na slučajan način mijenja poredak elemenata u listi.

1. Nasumično odabiranje čvorova – najjednostavnija izvedba Crossovog postupka utemeljena na Gauss-Seidelovoj metodi. Ovdje se za svaki iteracijski korak na slučajan način odabire čvor koji se uravnotežuje. Uvjet je da se ne odabire čvor koji je odabran u prethodnom koraku, kako bi se izbjegli koraci kojima se ne približavamo konačnom rješenju.

Pseudokod 1 Nasumično odabiranje čvorova

$lastVisitedNode \leftarrow undefined,\ \ error \leftarrow \infty,\ \ iteration \leftarrow 0$
while $iteration \leq MAX\_ITER$ && $error \geq TARGET\_ERROR$ **do**
    $nodeToVisit \leftarrow$ SHUFFLE($nodesForVisit$)[0]
    while $nodeToVisit == lastVisitedNode$ **do**
        $nodeToVisit \leftarrow$ SHUFFLE($nodesForVisit$)[0]
    end while
    RELAX_STEP($nodeToVisit$)
    $residualMoments \leftarrow$ CALCULATE_RESIDUALS( )
    $error \leftarrow$ ERROR_FUNCTION($residualMoments$)
    $iteration \leftarrow iteration + 1$
end while

2. Čvorovi s najvećim momentima – uvijek se posjećuju oni čvorovi u kojima je najveći neuravnoteženi moment. Dakle, uravnoteženje statičkog sustava se započinje u čvoru koji ima najveći moment izazvan zadanim opterećenjem. Nakon prve iteracije "titulu" najvećeg momenta preuzima neki drugi čvor te se njega u drugoj iteraciji uravnotežuje.

Pseudokod 2 Čvorovi s najvećim momentima

```
error ← ∞,  iteration ← 0
while iteration ≤ MAX_ITER && error ≥ TARGET_ERROR do
    nodeToVisit ← MAX(residualMoments)
    RELAX_STEP(nodeToVisit)
    residualMoments ← CALCULATE_RESIDUALS( )
    error ← ERROR_FUNCTION(residualMoments)
    iteration ← iteration + 1
end while
```

3. Isti redoslijed u svakom ciklusu – prvo se zadaje ili nasumično odabire redoslijed obilaženja čvorova. Redoslijed uzima u obzir svaki čvor jednom. Isti se ponavlja kroz sve cikluse obilaženja. Na taj način je osiguran jednak broj obilaženja svakog čvora.

Pseudokod 3 Isti redoslijed u svakom ciklusu

```
error ← ∞,  iteration ← 0
nodesForVisit ← SHUFFLE(nodesForVisit)
while iteration ≤ MAX_ITER && error ≥ TARGET_ERROR do
    indexToVisit ← iteration % LENGTH(nodesForVisit)
    nodeToVisit ← nodesForVisit[indexToVisit]
    RELAX_STEP(nodeToVisit)
    residualMoments ← CALCULATE_RESIDUALS( )
    error ← ERROR_FUNCTION(residualMoments)
    iteration ← iteration + 1
end while
```

4. Različiti redoslijed u svakom ciklusu – inačica je slična prethodnoj. Umjesto da se isti redoslijed provlači kroz sve cikluse, u svakoj će biti drugačiji. Dakle, za svaki ciklus uravnoteženja čvorova se nasumično odabire novi redoslijed. Pseudokod u odnosu na prethodnu inačicu ima malu preinaku.

Pseudokod 4 Različiti redoslijed u svakom ciklusu

```
error ← ∞,  iteration ← 0
while iteration ≤ MAX_ITER && error ≥ TARGET_ERROR do
    indexToVisit ← iteration % LENGTH(nodesForVisit)
    if indexToVisit == 0 then
        nodesForVisit ← SHUFFLE(nodesForVisit)
    end if
    nodeToVisit ← nodesForVisit[indexToVisit]
    RELAX_STEP(nodeToVisit)
    residualMoments ← CALCULATE_RESIDUALS( )
    error ← ERROR_FUNCTION(residualMoments)
    iteration ← iteration + 1
end while
```

5. **Istodobno obilaženje svih čvorova** – inačica u kojoj nema redoslijeda obilaženja čvorova. U svakom ciklusu obilaska čvorovi se uravnotežuju istodobno. U prethodnim načinima osnova je Gauss-Seidelova metoda dok je u ovom slučaju Jacobijeva. Stoga je algoritam malo drugačiji – dobivamo nove parametre tek nakon svakog ciklusa obilaženja umjesto svakog koraka (obilaska pojedinog čvora). Kako bi ova verzija bila usporediva glede broja iteracija s ostalim verzijama, broj izvedenih iteracija se množi s brojem čvorova.

Pseudokod 5 Istodobno obilaženje svih čvorova

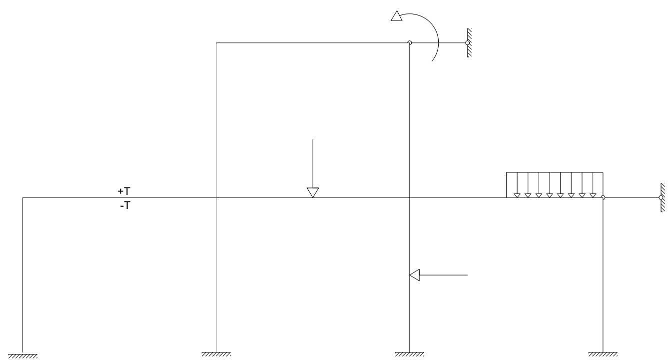
```
error ← ∞,  iteration ← 0
while iteration ≤ MAX_ITER && error ≥ TARGET_ERROR do
    indexToVisit ← iteration % LENGTH(nodesForVisit)
    nodeToVisit ← nodesForVisit[indexToVisit]
    RELAX_STEP(nodeToVisit)
    if indexToVisit ==LENGTH(nodesForVisit)−1 then
        residualMoments ← CALCULATE_RESIDUALS( )
        error ← ERROR_FUNCTION(residualMoments)
    end if
    iteration ← iteration + 1
end while
```

U svaki od odabranih 5. načina obilaska čvorova sustava se ulazi s istim uvjetima, ograničenjima i parametrima. Potrebni su početni momenti od zadanog opterećenja te razdjelni i prijenosni koeficijenti za svaki čvor. Za realizaciju u bilo kojem programskom jeziku može ih se zadati u obliku mapa ili lista. Nadalje, priloženi pseudokodovi predstavljaju algoritme koje je lako implementirati. Postupak će se izvoditi sve dok se ne postigne tražena točnost rješenja ili će stati u slučaju dostizanja zadanog broja iteracija. U konačnici je moguć ispis rezultata također u obliku lista ili pak grafova.

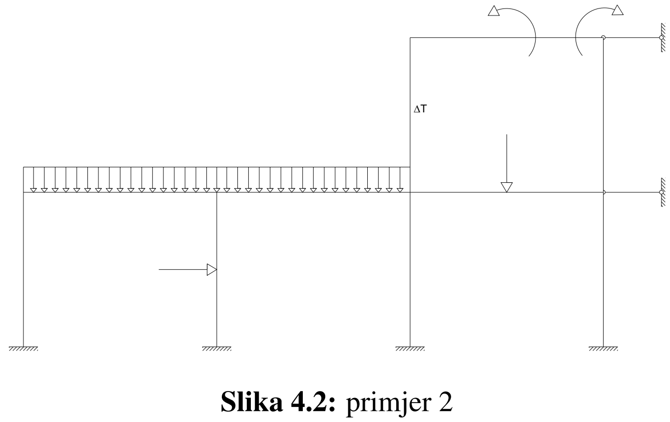
4. Primjeri i analiza broja računskih koraka

Kako bi se mogle uvidjeti prednosti i mane pojedinog načina obilaženja čvorova, treba ih primjeniti na konkretnim primjerima. Kod uravnotežavanja čvorova je najbitnije da se do zadovoljavajuće točnosti dođe sa što manje obilazaka odnosno sa što manje koraka iteracije. To je jako bitno pogotovo u slučaju ručnog rješavanja.

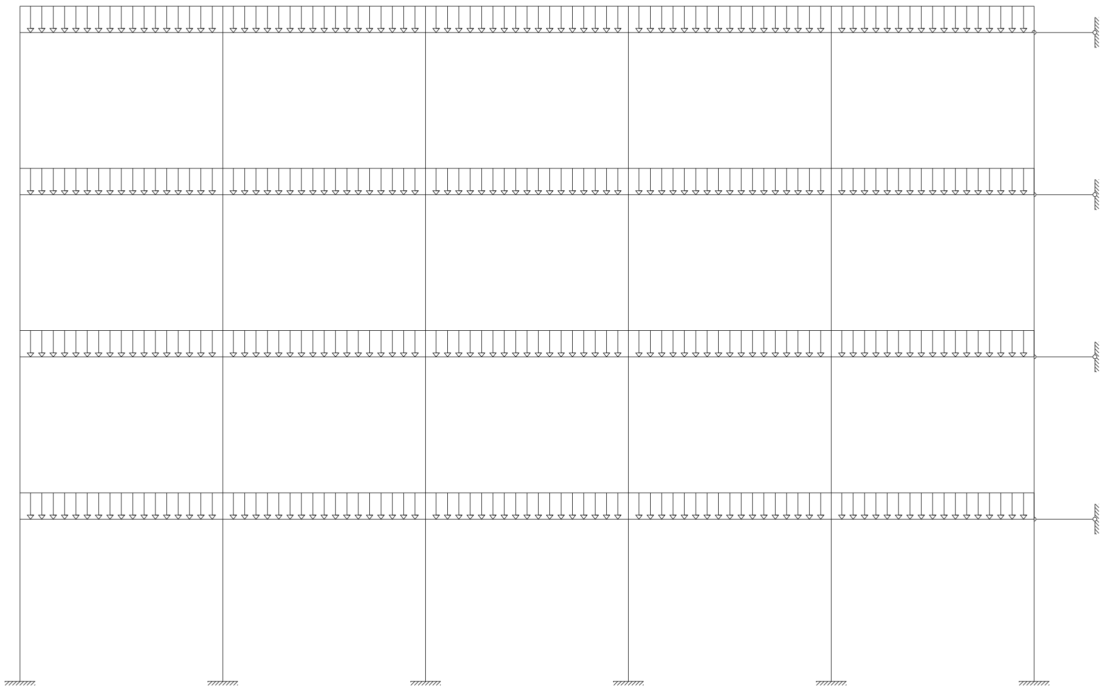
U programskom alatu *SageMath* su implementirane sve promatrane inačice na osnovi priloženih pseudokodova. Svakom inačicom je riješen svaki od tri sustava prikazanih na slikama 4.1, 4.2 i 4.3.



Slika 4.1: primjer 1



Slika 4.2: primjer 2



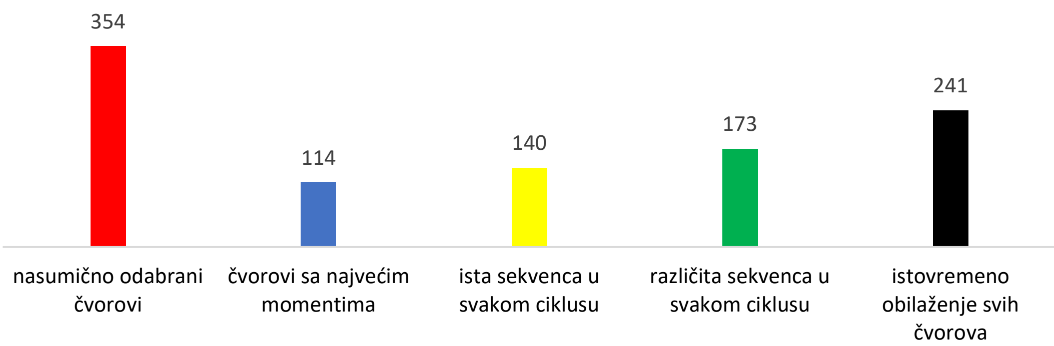
Slika 4.3: primjer 3

Postupci rješavanja inačica u kojima se nasumično odabire redoslijed su ponovljeni veći broj puta kako bi rezultati bili što uniformniji i što manje ovisni o samom nasumičnom odabiru. Svi postupci su odrađeni uz određene uvjete i ograničenja. U prvom slučaju uravnotežavanje sustava se odvijalo sve dok konačni momenti nisu odstupali od pravih konačnih momenata za zadanu vrijednost. U drugom slučaju uvjet prestanka postupka je bio zadani broj koraka iteracije.

Na slikama 4.4, 4.5 i 4.6 su prikazani potrebni brojevi iteracijskih koraka za dostizanje određene točnosti rješenja za svaki od primjera. Potreban broj iteracijskih koraka za svaki način obilaženja dobiven je za točnost rješenja reda veličine 10^{-2}. Iz priloženih histograma se vidi da neovisno o problemu koji se rješava, uvijek će iste inačice biti bolje od drugih. Lako je za uočiti da ako se obilaze čvorovi s najvećim momentima, do tražene točnosti rješenja se dolazi s najmanjim brojem koraka iteracije. No razlika između obilaženja čvorova s najvećim momentima i obilaženja ponavljajućim redoslijedom u slučaju opsežnijeg sustava nije tako velika, što se može vidjeti na slici 4.6. Nasumično obilaženje čvorova se pokazuje kao najlošiji izbor. Štoviše, može se uočiti da što je sustav složeniji, to je nasumično odabiranje čvorova, kao metoda, sve lošija. Dakle, za sustave s manje čvorova je daleko bolje obilaziti čvorove s najvećim momentima, no s povećanjem broja čvorova, relativna razlika među metodama, izuzev nasumične metode, je sve manja. Ovdje nije mjereno vrijeme izvođenja proračuna, nego samo broj koraka. Treba napomenuti da usložnjavanjem sustava nalaženje najvećeg reziduala postaje sve dugotrajnije.

Slika 4.4: primjer 1

Slika 4.5: primjer 2



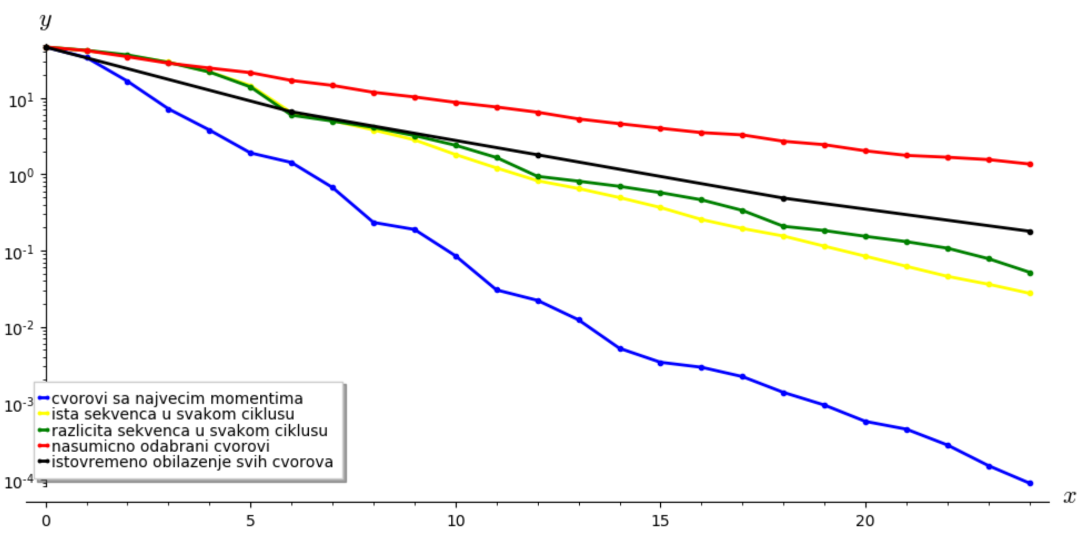
Slika 4.6: primjer 3

Definirana je funkcija greške

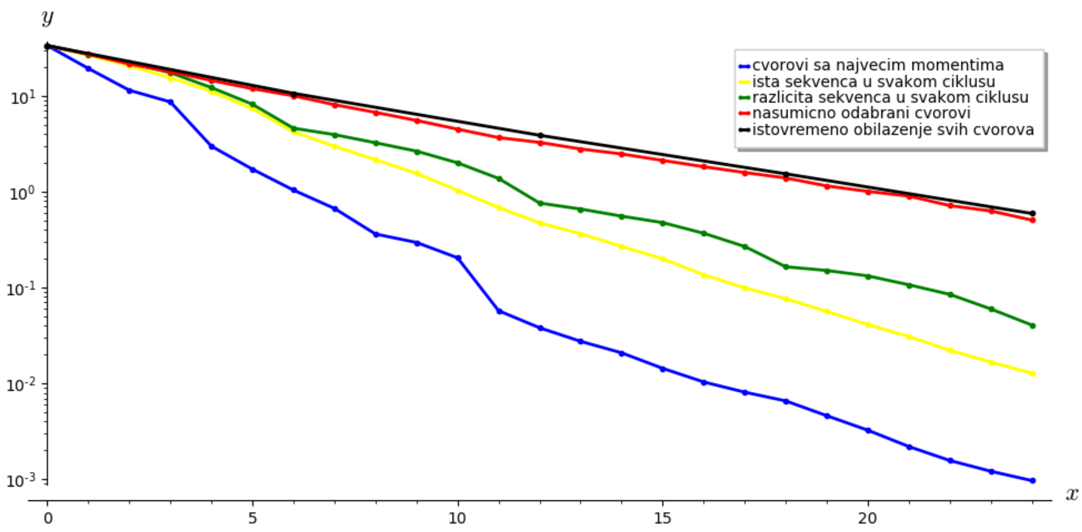
$$\varepsilon(k) = \frac{\sum |M_i^{(k)}|}{n}$$

za k-tu iteraciju, gdje sustav ima n čvorova. Svaka od pet promatranih inačica uravnotežavanja čvorova je opisana navedenom funkcijom. Kako bi se rezultati mogli zgodno usporediti, sve funkcije su prikazane na istim grafovima za zadane primjere (slike 4.7, 4.8 i 4.9). Na x-osi se nalazi broj koraka iteracije dok na y-osi iznos greške rješenja. Također, y-os je u logaritamskom mjerilu kako bi odnosi funkcija bili uočljiviji. Dakle, nakon svake iteracije je izračunata funkcija greške $\varepsilon(k)$ – srednja vrijednost rezidualnih momenata svakog čvora. Sve funkcije počinju iz iste točke. Razlog je to što, neovisno o inačici, rezidualni moment na početku je uvijek isti (za zadano opterećenje). U slučaju istovremenog obilaženja čvorova funkcija je računata tek nakon obilaska svih čvorova što je i vidljivo u grafovima.

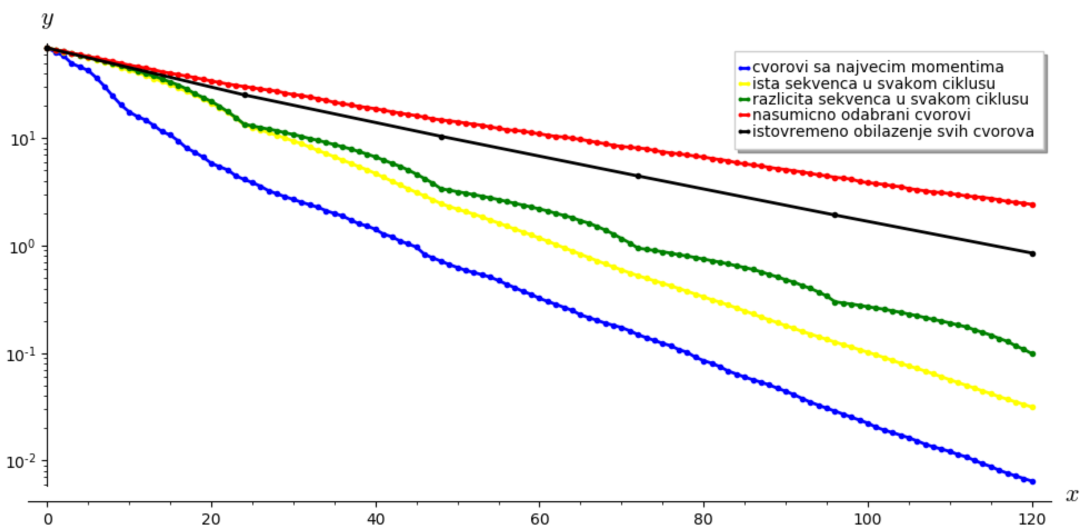
Opet se uočava kako su odnosi funkcija više-manje isti, neovisno o statičkom sustavu koji se uravnotežuje. Ističe se inačica u kojoj obilazimo čvorove s najvećim momentima. U svakom slučaju najbrže konvergira ka pravom rješenju. Iz priloženih grafova se vidi kako za isti broj iteracija daje rješenje s puno većom točnošću. Sljedeći način obilaska, poprilično učinkovit, je obilazak u ciklusima. Ako se redoslijed kroz cikluse obilaženja ne mijenja, dobit će se bolji rezultati nego kad bi se za svaki ciklus generirala nova sekvenca. Iz grafa na slici 4.9 se vidi kako se "prednost" funkcije za obilaženje čvorova s najvećim momentima ne povećava nakon određenog broja iteracija. Stoga se za opsežnije statičke sustave slobodno može upotrebljavati i obilazak redoslijedom. Nadalje, ako se čvorovi obilaze slučajnim odabirom, neće se postići praktično rješenje za prihvatljiv broj koraka iteracije. Doduše, zbog konvergencije, kad-tad će se doći do rješenja, ali pokraj ostalih inačica nema ju smisla prakticirati.



Slika 4.7: primjer 1



Slika 4.8: primjer 2



Slika 4.9: primjer 3

Među lošijim (pa i najlošijom (slika 4.8)) pokazala se i inačica u kojoj istodobno uravnotežujemo sve čvorove. Također i ona neučinkovito konvergira kao i nasumični odabir čvorova. No ne treba ju potpuno otpisati jer ima ona, u nekim okolnostima, i svoje prednosti koje su spomenute u zaključku.

5. Zaključak

Rješavanje statičkog sustava je moguće na popriličan broj načina. Jedan od njih je Crossov način. Pokazano je kako ni Crossova metoda nema jedinstveni slijed proračuna. Iteracijski dio postupka u ovom radu je odrađen na pet različitih načina. Primijenivši inačice na različitim primjerima slijedili su rezultati u obliku histograma i grafova. Na osnovi broja iteracija, brzini konvergencije i funkciji pogreške napravljena je analiza.

Ustanovljeno je da je inačica u kojoj obilazimo čvorove s najvećim momentima najbolja u svakom pogledu. Primjenjive su i inačice u kojima čvorove obilazimo u ciklusima. Dobiveni rezultati su za slučajan odabir redoslijeda u ciklusu. Ako se može naslutiti povoljan redoslijed, on se može i izravno zadati, pa na taj način poboljšati dobivene rezultate. Preostale dvije metode već iziskuju previše koraka iteracije za precizno rješenje. U odnosu na ostale inačica sporo konvergiraju. Metoda slučajnog odabira je vrlo jednostavna, pa joj to ide u korist. Također i u slučaju istodobnog obilaska čvorova do rješenja se dolazi na jednostavan način. Ova inačica je zanimljiva za paralelna računala u kojima se svakom čvoru može dodjeliti "vlastiti" procesor, pa se uravnoteženje svih čvorova zaista izvršava istodobno. Isto tako, ako nam je potrebno okvirno poznavanje rješenja, na taj se način, ručno, u nekoliko jednostavnih iteracijskih koraka dolazi do praktično primjenjivog rješenja.

Dakle, u slučaju da sustav uravnotežujemo kako bismo dobili konačne vrijednosti momenata, najbolje je obilaziti čvorove s najvećim neuravnoteženim momentima. Konačni momenti se dobivaju brzo i zanemarivo odstupaju od pravih vrijednosti momenata te se mogu upotrijebiti za daljnje proračune dimenzioniranja.

LITERATURA

[1] T. Došlić. *Numerička matematika, skripta*. Građevinski fakultet Sveučilišta u Zagrebu, 2014.

[2] K. Fresl. *Građevna statika 2. Predavanja, skripta*. Građevinski fakultet Sveučilišta u Zagrebu, 2017. URL `http://grad.hr/nastava/gs/gs2/gs2.pdf`.

[3] S. Liu. *On Convergence of the Hardy Cross Method of Moment Distribution*. A thesis submitted to the Graduate Faculty of North Carolina State University in partial fulfillment of the requirements for the Degree of Master of Science, 2015.

[4] Y. Saad. *Iterative methods for sparse linear systems, 2nd edition*. Society for Industrial and Applied Mathematics, Philadelphia, 2003.

[5] S. Suljagić. *Matematika III*. Zagreb, 2001. URL `http://www.grad.hr/nastava/matematika/mat3/index.html`.

[6] Sage Tutorial v9.1, 2020. URL `https://doc.sagemath.org/html/en/tutorial/`.

Inačice Crossova postupka

Sažetak

U radu je obrađen jedan od relaksacijskih postupaka – Crossov postupak. U prvom dijelu je opis postupka, izvedeni su svi potrebni izrazi te je postupak dodatno pojašnjen kroz primjer. Nadalje su objašnjeni Jacobijev i Gauss-Seidelov iteracijski postupak te je ostvarena poveznica s Crossovim postupkom. Naglasak rada je na inačicama Crossova postupka. Uz pomoć programskog alata $SageMath$ ispitane su sve inačice na istim primjerima statičkih sustava. Konačno, napravljena je analiza računskih koraka kao i sama usporedba inačica.

Ključne riječi: Crossov postupak, Jacobijev postupak, Gauss-Seidelov postupak, inačice, redoslijedi obilaženja čvorova, iteracijski postupak, relaksacijski postupak

Variants of Cross method

Abstract

This thesis deals with one of the relaxation methods – the Cross method. The first part describes the method, all the necessary equations are derived and the method is further explained through an example. Furthermore, the Jacobi and Gauss-Seidel iterative methods are explained and a relation with Cross method is made. The emphasis of the thesis is on the variants of Cross method. With $SageMath$ software, all variants were tested on the same examples of static systems. Finally, an analysis of the calculation steps is made, as well as the comparison of the variants.

Keywords: Cross method, Jacobi method, Gauss-Seidel method, versions, node traversal sequences, iterative method, relaxation method