Факултет инжењерских наука Универзитета у Крагујевцу

Марко Љ. Гицић

Параметарска оптимизација решеткастих конструкција коришћењем API програмирања

завршни рад

Крагујевац, 2020.

Факултет инжењерских наука Универзитета у Крагујевцу

Назив студијског програма: Машинско инжењерство

Ниво студија: Основне академске студије

Модул: Информатика у инжењерству

Предмет: Рачунарски алати

Број индекса: 197/2016

Марко Љ. Гицић

Параметарска оптимизација решеткастих конструкција коришћењем API програмирања

завршни рад

Комисија за преглед и одбрану:

1. **Др Владимир П.Миловановић, доцент - ментор**
2. _____ Датум одбране:_____
3. _____ Оцена:_____

Универзитет у Крагујевцу

Факултет инжењерских наука

Основне студије: **Машинско инжењерство**

Студијско подручје (усмерење)**: Информатика у инжењерству**

Име и презиме: **Марко Гицић**

Број индекса: **197/2016**

ПРИЈАВА ЗАВРШНОГ РАДА

Тема рада: **Параметарска оптимизација решеткастих конструкција коришћењем API програмирања**

Задатак: У оквиру завршног рада кандидат треба да прикаже поступак параметарске оптимизације решеткастих конструкција коришћењем софтвреског пакета FEMAP with NX Nastran и API програмирања. За изабрани попречни пресек решеткасте конструкције од одговарајућег материјала са прописаним степеном сигурности, потребно је направити аутоматски поступак параметаске оптимизације где би као крајњи излаз требало добити оптимизовану решеткасту конструкцију најмање могуће масе, која у потпуности задовољава прописане критеријуме. На основу свих резултата до којих је кандидат дошао приликом израде завршног рада потребно је извести закључна разматрања и дефинисати правце будућих истраживања

Завршни рад треба да садржи следеће целине:

1. Уводна разматрања
2. Појам и област примене решеткастих конструкција
3. Основни појмови API програмирања
4. Алгоритам за параметарску оптимизацију
5. Параметарска оптимизација изабране решеткасте конструкције
6. Закључак
7. Литература

Ментор:

др Владимир П. Миловановић, доцент

Крагујевац, 17.08.2020.

Резиме

Идеја овог завршног рада је да се прикаже поступак параметарске оптимизације решеткастих конструкција коришћењем софтверског пакета Femap и API програмирања. За кружни попречни пресек профила решеткасте конструкције од одговарајућег материјала и степена сигурности потребно је направити аутоматизовани поступак оптимизације. Приликом овог поступка потребно је задовољити одређене критеријуме а да конструкција буде што мање масе. У првом делу рада дат је појам и област примене решеткастих конструкција.

У другом делу рада су приказани основни појмови везани за API програмирање као и неке од основних функција и метода. Приказани су објекти као и типови података који су заступљени у API програмирању.

У трећем делу је приказан алгоритам за параметарску оптимизацију решеткастих коснтрукција коришћењем методе коначних елемената.

У четвртом делу је одрађен поступак позивања API апликације у Femap-у која има задатак да изврши параметарску оптимизацију, као и приказ масе конструкције пре и после оптимизације.

На крају рада дати су неки од битних закључака до којих је дошло приликом израде завршног рада.

Кључне речи: метода коначних елемената, решеткаста конструкција, оптимизација, API програмирање

Summary

The main idea of the final paper is to present procedure of parametric optimization of the lattice constructions using software Femap with API programming. For circular cross section of profile lattice construction from appropriat material and degree of security is required to do an automated optimization process. During this procedure certain criteria need to be met and keep the structue lightes as possible. In the first part of this paper it was given the concept and field of application of lattic structures.

In the second part basic concepts related to API programming and some of the basic functions and methods are presented. Some of the objects and data type that are represented in API programming are shown.

In the third part is presented an algorithm for parametric optimization of lattice structures using finite element method.

The fourth pard describe the procedure of calling an API application in Femap, which has the task of parametric optimization, as well and showing the mass of the structure before and after optimization.

At the end of the paper some of the important conclusions that wher reached during the final paper are given.

Key words: finite element method, lattice construction, optimization, API programming

Садржај

1 Уводна разматрања

Конструкција која је сачињена од правих штапова спојених на крајевима тако да она образује непроменљив систем назива се решетка. Решеткасте конструкције су настале из потребе да се смањи утрошак материјала уз могућност максималног искоришћења носивости конструкције. Ово максимално искоришћење конструкције постиже се под претпоставком да је решеткасти конструкциони систем геометријски непроменљив.

Под појмом „решавање решетке“ се подразумева одређивање одговарајућих реакција веза и унутрашњих сила у штаповима при задатим спољашњим активним силама и оптерећењима [1].

Стандардне методе решавања решеткастих конструкција се могу поделити на: аналитичке, графо-аналитчке и графичке методе. Основа аналитичких метода је у томе да се постављају једначине равнотеже које нам служе за одређивање непознатих. Графичке методе се темеље на истим принципима као и аналитичке, односно може се рећи да су геометријске интерпретације једначина равнотеже. Неке од методе решавања раванске решетке су [1]:

- метода изрезивања чворова;
- Кремонина метода;
- метода пресека решетке као целине.

Метода изрезивања чворова којом се тврди да је сваки изрезани чвор у равнотежи под утицајем свих сила које на њега делују (спољашњих активних, спољашњих реакција веза и сила у штаповима).

Графичка метода изрезивања чворова је веома једноставна и сасвим прегледна. Недостатак јој је што се све унутрашње силе штапова решетке уј затвореним полигонима сила појављују два пута. Овај недостатак нарочито долази до изражаја код решетки са великим бројем штапова. У циљу превазилажења овог недостатка настала је Кремонина метода, скупе се заједно сви затворени полигони сила чворова тако да се добије један полигон у којем би се свака унутрашња сила штапа појавила само једном. Такав полигон се назива Кремонин план сила.

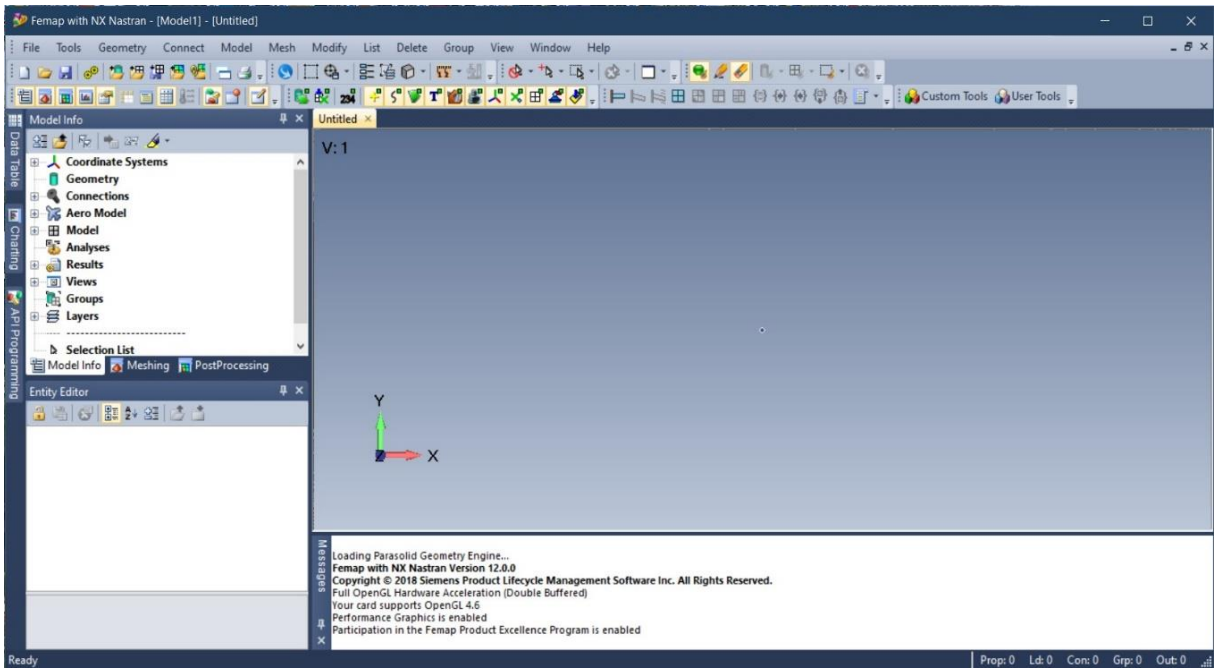
У методе пресека спада Куламнова метода. Она се заснива на могућности уравнотежења једне задате силе са 3 силе чији су правци деловања познати а величину и смер им треба одредити. За одређивање сила у штповима решетке помоћу пресека користи се и Ритерова аналитичка метода. Ова метода представља примену аналитичких услова равнотеже помоћу сума момената за три тачке које не леже на истој правој већ на теменима троугла.

Поред поменутих „традиционалних“ метода за решавање решеткастих конструкција у данашње време све више се користе и нумеричке методе. Метод коначних

елемената (МКЕ) као најопштији метод дискретне анализе представља једну од метода која се са великим успехом може користити за решавање раванских решетки.

Метод коначних елемената (МКЕ) је данас најопштији нумерички метод, примењен у готово свим наукама, а посебно у инжењерским областима. Сматра се да је метод коначних елемената метод који је, у поређењу са другим, имао највећи утицај на развој и домете техничких дисциплина. Основне карактеристике методе коначних елемената које га доводе испред других метода су састоје у његовој општости, применљивости на линеарне и нелинеарне проблеме, као и погодности примене рачунара. Применљив је на област солида, флуида, провођења топлоте и уопштено на проблеме поља физичких величина као и на спрегнуте проблеме. Развој методе коначних елемената је директно везан за појаву рачунара. Јер, практична примена методе коначних елемената, поред осталог, подразумева решавање великог система алгебарских једначина, чији број даље иде на више десетина хиљада једначина [2].

Софтверски пакет FEMAP (Finite Element Modeling And Postprocessing) је програм за потребе различитих врста инжењерских анализа развијен од стране компаније Siemens PLM Software. Користи се за пре и пост процесирање различитих врста конструкција, делова, склопова или подсклопова методом коначних елемената. У FEMAP се могу импортовати читаве геометрије у различитим форматима, које даље служе за креирање одговарајућих МКЕ модела припремљених за даље анализе. FEMAP омогућава поспроцесирање добијених резултата анализе добијених различитим МКЕ софтверима [3].



Слика 1.1 Femap радно окружење

Све ово напред наведено нам говори да се FEMAP може комбиновати са различитим CAD системима и програмима који раде на основама методе коначних елемената. У склопу FEMAP -а се налазе интегрисани „солвери“, чија се решења могу са великим успехом постпроцесирати и даље користити за различите врсте даљих анализа. Радно окружење софтверског пакета FEMAP приказано је на слици 1.1.

У саставу програмског пакета FEMAP налази се интегрисан интерфејс за API програмирање. Апликациони програмски интерфејс (Application Program Interface), познат и као API, је интерфејс за програмирање који дефинише начине на које апликације могу да прузму услуге од библиотека или оперативног система. API омогућава додатну функционалност FEMAP -а користећи његове алате, особине и податке. То је самосталан програм који интерагује са FEMAP-ом.

Оптимизација се дефинише као научна дисциплина која се бави одређивањем најбољег решења одређеног, математички дефинисаног, проблема. Поседовање најбољег решења или решења блиских најбољем води ка уштедама у материјалу, енергији, „новцу“ или постизању највеће поузданости или сигурности у раду. Оптимизацијом се тежи минимизација негативних ефеката на првом месту трошкови или максимизација позитивних ефеката (добит) [4]. Поступак решавања оптимизационих задатака се састоји од: формулације проблема; израде математичког модела који представља реални систем; избора и примене одговарајуће методе; избора алгоритма и програма за рачунар; тестирања модела и добијених решења и имплементације.

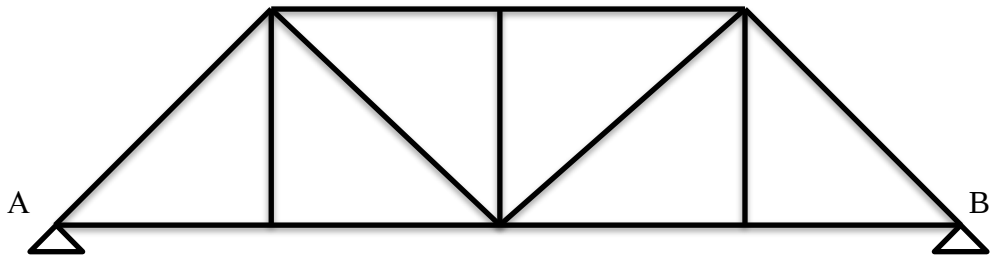
Конструкционе променљиве код решеткастих носача су: дужина и распоред њених појединачних делова (штапова) и димензије попречног пресека. Циљ оптимизације је да комбинације конструкционих променљивих дају најлакшу решеткасту конструкцију а да при томе буду задовољена прописани критеријуми. За потребе параметарске оптимизације пројектоване решеткасте конструкције, код које је сваки члан (штап) одређене коначне дужине и положаја потребно је одредити минималну вредност попречног пресека.

Коришћењем МКЕ, софтверског пакета FEMAP са интегрисаним МКЕ солвером NX Nastran и API програмирања и њиховим спајањем у једну целину желимо да аутоматизујемо процес параметарске оптимизације раванске решеткасте конструкције.

Како би се цео процес оптимизације убрзао у односу на ручну оптимизацију (ручно мењање параметара) и избегли редудантни кораци који би се појављивали, односно како би се процес оптимизације аутоматизовао написан је API програм за параметарску оптимизацију решеткасте конструкције.

2 Појам и област примене решеткастих конструкција

Као резултат жеље да се добију носачи уз минимални утрошак материјала, настале су решеткасте конструкције. Решеткасти носачи се у односу на пуне носаче одликују смањеном тежином. Конструкција која је сачињена од правих штапова спојених на крајевима тако да она образује непроменљив систем се обично зове решетка или решеткаста структура [1]. Пример једне решеткасте конструкције је приказан на слици 2.1.



Слика 2.1 Решеткаста конструкција

Решеткасти носачи са становишта утрошка материјала су повољнији од саћастих и пуних носача. Састоје се од међусобно повезаних појасних штапова и штапова испуне (у правцу дијагонале и вертикале) који формирају троугаону структуру. Основна претпоставка која се односи на решеткасте носаче односно чланове решетке, структурне штапове, је да су оптерећени увек само са две силе, на местима везивања. Постизање бољег искоришћења напона код решеткастих конструкција се постиже тиме што су штапови аксијално напрегнути па је дијаграм нормалних напона константан. Момент савијања код решеткастих конструкција преноси се напрезањем појасних штапова. Утицај трансферзалних сила прихватају штапови испуне.

Уколико неки систем сила делује на три штапа зглобно везана на крајевима, услед утицаја сила облик троугла занемарљиво мало се мења па се може сматрати да је оваква фигура непроменљива – крута структура. Равanska решетка ће бити непроменљив систем ако је образована само од троуглова.

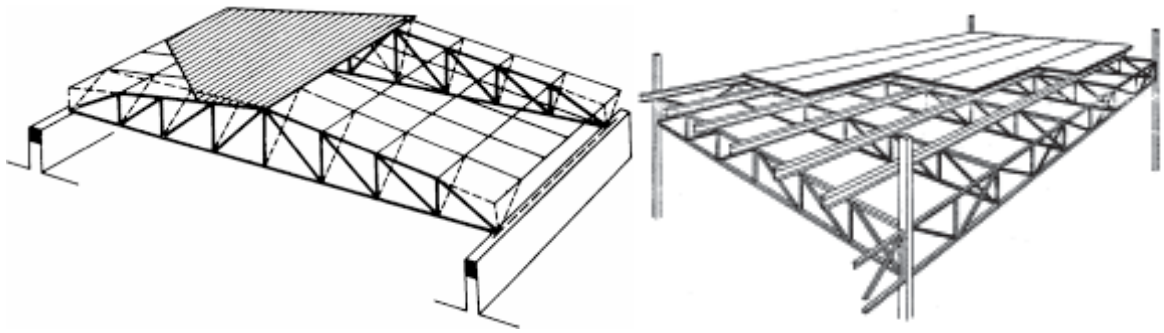
Решеткасте конструкције углавном се примењују за пријем тешких оптерећења и премошћавање већих распона [1]. Предности решеткастих носача у односу на пуне носаче су следећи: боље искоришћење материјала у односу на пуне носаче, мања тежина, решеткасти носачи пропуштају више светлости; док је један од недостатака њихова комплексност, потребан је већи број радних операција за њихову израду, што утиче на цену решеткастих носача.

У металним конструкцијама решеткасти носачи се веома често користе и то како у зградарству тако и у мостоградњи. Бројни су примери примене решеткастих носача. У зградарству се користе као: рожњаче, кровни носачи, подвлаке, подни носачи, крански носачи у индустријским халама, спрегови итд.

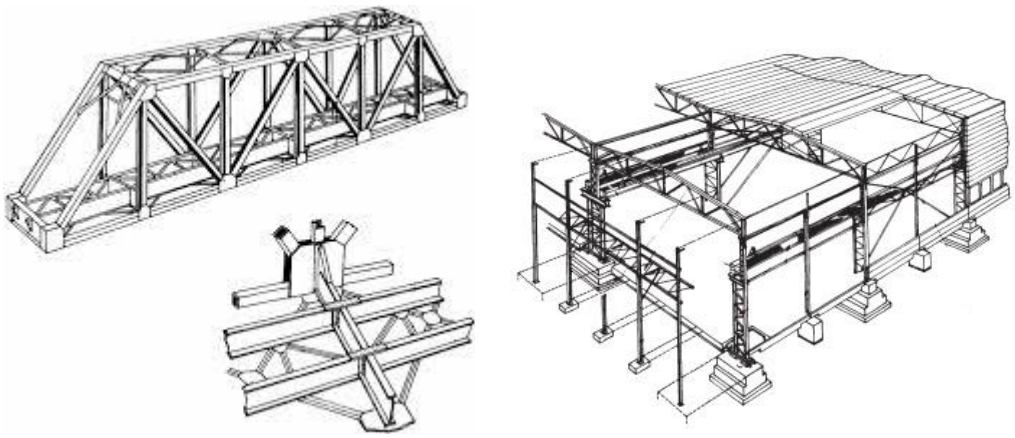
Кровни носачи у решеткастој изради примењују се у готово свим типовима објеката, од индустријских хала, преко објеката високо градње до спортских и конгресних дворана и изложбених павиљона. Избором облика решеткасте структуре и попречних пресека штапова могу се добити решеткасти носачи веома атрактивног изгледа, тако да челична конструкција постаје саставни део ентеријера.

Решеткасти подни носачи се примењују када је потребан слободан простор без стубова. Посебно су погодни јер омогућавају провођење инсталационих цеви између штапова испуне, па се на тај начин избегава повећање висине међуспратне конструкције.

У мостоградњи се решеткасти носачи користе као главни носачи код друмских, железничких и транспортних мостова у индустријским објектима. Осим тога, као и у зградарству и у мостоградњи је веома честа примена спрегова за стабилизацију, којима се обезбеђује пријем хоризонталних сила (ветра, сеизмика,...) и просторна стабилност конструкције.



а) б)



в) г)

Слика 2.2 Примери примене решеткаста конструкција [5]

На слици 2.2 приказано је неколико карактеристичних решеткастих конструкција. Кровни носач карактеристичан за објекте високоградње дат је на слици 2.2.а. Пример међуспратне конструкције који сачињавају решеткасти подни носачи и подвлаке дат је на слици 2.2.б. На слици 2.2.в и 2.3 приказана је решеткаста конструкција железничког моста са решеткастим главним носачима. На слици 2.2.г приказана је једна индустријска хала са решеткастим кровним носачима и стубовима. Решеткаста конструкција крана приказана је на слици 2.4.

Слика 2.3 Решеткаста конструкција моста [6]

Слика 2.4 Решеткаста конструкција крана [7]

3 Основни појмови API програмирања

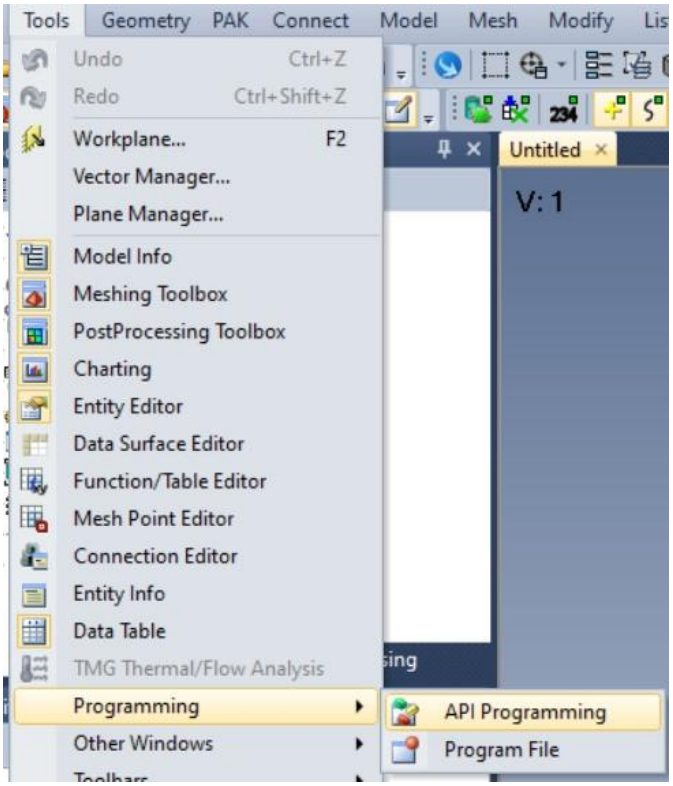
Апликациони програмски интерфејс (Application Program Interface), познат и као API, је интерфејс за програмирање који дефинише начине на које апликације могу да користе услуге од библиотека или оперативног система. API може пружити механизме проширења како би могли проширити постојећу функционалност на различите начине и у различитом степену. Може бити у потпуности прилагођен и омогућује кориснику да користи интерфејс независно од имплементације.

API је обично повезан са софтверском библиотеком. API описује и прописује „очекивано понашање“ (спецификацију), док је библиотека „стварна имплементација“ овог скупа правила. Један API може имати више имплементација (или ниједну, апстрактну) у облику различитих библиотека које деле исти програмски интерфејс. Одвајање API-ја од његове имплементације може омогућити програмима написаним у једном језику да користе библиотеку написану на другом. API може одредити интерфејс између апликације и оперативног система.

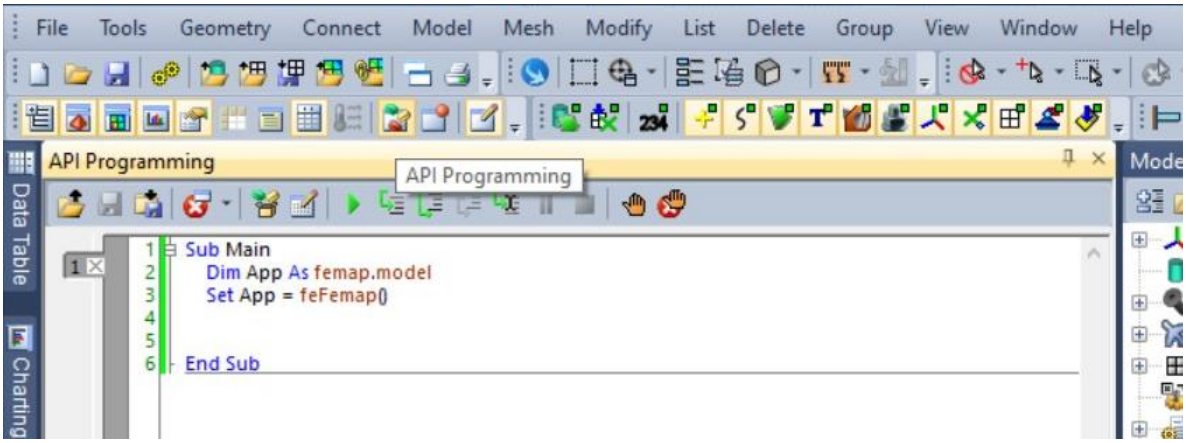
Назив API може се односити на језике и библиотеке које могу да контролишу FEMAP, али се такође односи и на засебне API програме. FEMAP API нам омогућава да контролишемо FEMAP без отварања алатки или ручног уноса података, API нам омогућава обављање одређеног задатка унутар FEMAP-а или неког другог програма попут Excel-а. FEMAP API је базиран на Microsoft-овом OLE/COM програмском интерфејсу [5].

OLE је скраћеница од Object Linking and Embedding и представља механизам који дозвољава корисницима да креирају и измењују постојеће „објекте“ креираних од стране више апликација [8]. Језик који се најчешће употребљава у FEMAP API програмирању је Visual Basic, мада постоји могућност коришћења и других објектно оријентисаних језика попут Python-а и C#, где за сваку команду у FEMAP-у постоји и API команда.

Најлакши начин за писање FEMAP API програма је отварање прозора API Programming који се налази у софтверском пакету FEMAP. Приступ овом прозору могућ је избором команди ***Tools/ Programming/ API programming*** (слика 3.1) при чему се отвара прозор приказан на слици 3.2.

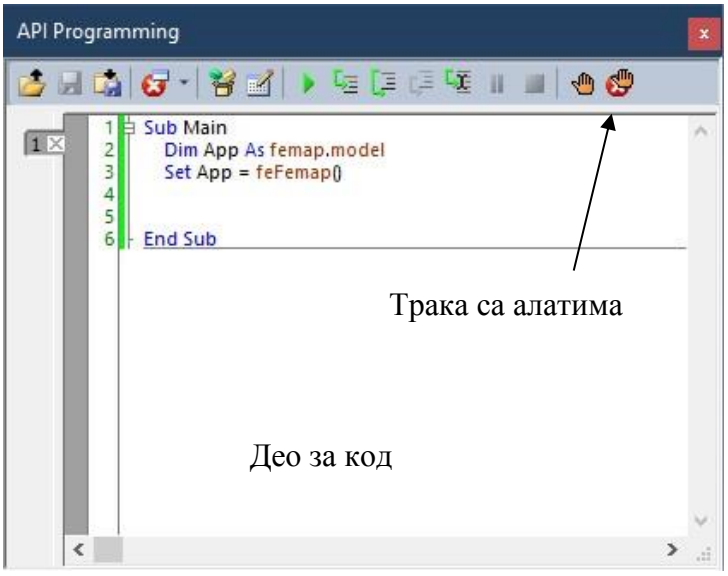


Слика 3.1 Команде за приступ API Programming прозору



Слика 3.2 API programming прозор

У FEMAP-овом интегрисаном интерфејсу за програмирање користи се WinWrap® Basic који спада у подгрупу Visual Basic-а. Омогућава писање, измене, „дебагинг“ и покретање апликација у језику веома сличном Visual Basic-у. На слици 3.3 приказан је FEMAP-ов интегрисан интерфејс за програмирање. У траци са алатима се налазе команде за отварање већ постојећих API програма, чување, креирање нових, покретање програма, паузирање и заустављање програма.



Слика 3.3 – API programming прозор

Елементи API програмирања спадају у једну од три категорије:

1. Апликациони објекти
   - Глобалне промењиве и константе
   - Методе
2. Објекти ентитета
   - Ентитети модела (чворови, елементи, својства и караактристике , излазни вектори, итд.)
3. Објекти алата
   - Помажу при креирању и изменама ентитета објеката
   - Табела података
   - Сетови, итд.

3.1 Апликациони објекти

Први ниво укључује FEMAP апликациони објекат. У свакој апликацији која се напише, увек се прво дефинише и референцира FEMAP апликациони објекат, који формира везу између написане апликације и FEMAP-а.

Апликациони објекат садржи својства (податке) и методе (функционалност) које су глобалне по природи. Прво је потребно доделити приступ FEMAP API-ју и приступ FEMAP моделу. На овај начин се омогућава приступ свим другим методама и својствима. Приликом дефинисања FEMAP објекта потребно га је дефинисати као глобални објекат, тако да приликом коришћења функција „get“ или „create“ буде доступан током читавог рада написане апликације.

Како би се повезала апликација са FEMAP-ом, потребно је декларисати објекат који ће омогућити везу. Успостављање везе је могуће на два начина помоћу функције `GetObject` уколико апликацију повезујемо са FEMAP сесијом која се већ одвија или уз помоћ функције CreateObject ако је потребно покренути нову FEMAP сесију.

Следећи пример приказује везу успостављену функцијом `GetObject`, уколико је потребно повезивање са већ покренутом FEMAP сесијом.

```
Dim App As Object
Set App = GetObject(, "femap.model")
```

Ако желимо да покренемо нову FEMAP сесију, која неће бити приказана, користимо следећи приступ:

```
Dim App As Object
Set App = CreateObject("femap.model")
```

Оваквим приступом након изласка из FEMAP-а апликација ће и даље радити у позадини. Како би се ово избегло потребно је доделити променљиви femap вредност:

```
femap = nothing
```

Ако је отворено више модела (више радних прозора) у истом FEMAP-у, API апликација увек ће успостављати везу са прво отвореним моделом ако се користи функција `GetObject`. Ово представља недостатак OLE/COM програмског интерфејса и не постоји могућност заобилажења овог недостатка коришћењем функције `GetObject`.

Уколико се користи интегрисано WinWrap радно окружење (API Programming) постоји могућност заобилажења поменутог недостатка, повезивања са већ постојећом FEMAP сесијом без коришћења `GetObject` функције:

```
Dim App As femap.model
Set App = feFemap()
```

Која се аутоматски генерише при покретању WinWrap радног окружења. Функција `feFemap()` ради исто као и функција `GetObject()`, предност ове функције је да може да одреди ако је отворено више модела у истом FEMAP-у са којом сесијом треба да се повеже.

3.2 Објекти ентитета

Објекти ентитета омогућавају креирање упита и манипулацију подацима модела, односно са свим што је креирано или сачувано у FEMAP-овој бази података. Ови објекти омогућавају детаљну контролу која је често потребна за имплементацију у функцијама. Под ентитете спадају чворови, елементи, материјал, подешавање анализе и слично. Како би се креирао објекат прво је потребно декларисати објекат, онда користимо једну од

FEMAP-ових функција за креирање објекта. Када је објекат креиран, сва својства и методе за тај тип објекта постају дозвољене за коришћење.

На следећем примеру показан је поступак креирања објекта. У примеру се креира објекат типа чвор, приступа чвору 2 и врши се замена његове X координате, и нова локација се чува у моделу.

```
Sub Main
	Dim App As femap.model
	Set App = GetObject(, "femap.model")
	Dim nd As Object
	Set nd = App.feNode
	rc = nd.Get(2)
	nd.x = 70
	rc = nd.Put(2)
End Sub
```

У табели 3.1 приказане су неке од метода које омогућавају манипулацију ентитетима.

Табела 3.1. Методе објекта ентитета

Метода	**Опис методе**
`feGroup`	Креира објекат типа Group
`feLayer`	Креира објекат типа Layer
`feLoadDefinition`	Креира објекат типа LoadDefinition
`feLoadSet`	Креира објекат типа LoadSet
`fePoint`	Креира објекат типа Point
`feSolid`	Креира објекат типа Solid/Volume
`feConnection`	Креира објекат типа Connection
`feElem`	Креира објекат типа Element
`feProp`	Креира објекат типа Property
`feNode`	Креира објекат типа Node
`feText`	Креира објекат типа Text

Објекти ентитета су објекти који садрже податке који су делови FEMAP модела. Ово укључује чворове, елементе, материјале, својства, тачке, линије, површине и запремине (солиде). Већина метода се бави преласком ентитета у модел или преузимањем или складиштењем података ентитета.

Неки од објека ентитета су: `AnalysisCase Objects`, `AnalysisMgr Objects`, `BCGeomObjects`, `BCNode Objects`, `Chart Objects`, `Curve`

`Object Methods`, `Group Objects`, `Element Objects`, `Layer Objects`, `LoadSet Objects`, `Material Objects`, `Node Objects`, `Property Objects` итд. У табели 3.2 су приказане неке од заједничних метода за наведене објекте ентитета.

Табела 3.2. Заједничке методе објекта ентитета

Метода	**Опис методе**
`Get(id)`	Преузима ентитет из базе података модела са одређеним id-ом
`Put(id)`	Чува ентитет у бази података модела са одређеним id-ом
`Exist(id)`	Проверава да ли постоји одређени елемент
`Deletable(id)`	Проверава да ли је ентитет могуће обрисати
`Delete(id)`	Брисање ентитета из базе података
`Next(void)`	Преузима следећи ентитет са већим ид-ом
`NextEmptyID(void)`	Проналази први већи непостојећи ид
`SelectID(title)`	Приказује дијалог како би омогућило кориснику да одабере један ентитет одређеног типа

AnalysisCase објекти одговарају случајевима и сетовима за анализу. Користи се `feAnalysisCase` метода за креирање ових објеката. AnalysisMgr објекти одговарају сетовима анализе у моделу. За креирање ових објеката се користи `feAnalysisMgr`. BCGeom објекти одговарају ограничењима базираних на геометрији. За креирање ових објеката користи се `feBCGeom`. BCNode објекти одговарају чворним ограничењима, креирају се коришћењем `feBCNode` методе. Chart објекти одговарају табелама у делу са табелама, креирају се коришћењем `feChart` методе. Curve објекти одговарају линијама, користи се `feCurve` за креирање објекта. Објекти типа Group одговарају групама у моделу. Element објекти су повезани са елементима у моделу, за њихово креирање користи се `feElem` метода. Layer објекти одговарају дефинисаним лејерима у моделу, за креирање се користи `feLayer` метода. LoadSet су повезани са сетовима оптерећења, користи се `feLoadSet` метода. Material објекти одговарају материјалима у моделу, користи се `feMatl` метода. Објекти Node су повезани са чворовима у моделу. Креирају се коришћењем `feNode` методе. Property објекти одговарају својствима у моделу. Коришћењем `feProp` методе или Prop методе објеката елемената креира се објекат типа Property. Ово су само неке од многобројних објеката који се налазе у целини објекта ентитета. Сви ови објекти су изведени из објеката ентитета, тако да поред својства и метода које су овде описане сва својства и методе која важе за објекте ентитета доступне су и за све остале објекте.

3.3 Објекти алата

Поред апликационих и објекта ентитета FEMAP нуди и друге објекте који омогућавају приступ другим функционалностима и алатима. Неки од ових објеката

омогућавају приступ разним прозорима доступним у корисничком интерфејсу, други селекцију ентитета и приступ подацима и функцијама. Како би се направио нови објекат алата потребно је прво декларисати објекат, након чега се користи једна од наведених функција. у табели 3.3, приказане су неке од функција у оквиру објекта алата.

Табела 3.3. Методе објекта алата

Метода	**Опис методе**
`feDataTable`	Креира објекат типа DataTable
`feRead`	Креира објекат типа ReadFile
`feSelector`	Креира објекат типа Selector
`feSet`	Креира објекат типа Set
`feCopyTool`	Креира објекат типа CopyTool
`feMoveTool`	Креира објекат типа MoveTool

DataTable објекат омогућује приступ табели података. У већини метода тражи се да се спецификују индекси редова и колона података над којим се врши манипулација. За приступ DataTable објектима користи се `feDataTable` метода. Неке од метода су `FindRow` која служи за налажење одређеног реда, враћа индекс траженог реда. Ентитети који се налазе уз ову методу су (`ModelID, SetID, ID`). `GetRowInfo` која враћа податке свих редова у табели, `FindColumn` налажење колоне, `GetColumnInfo` враћа податке свих колона у табели, `GetColumnWidth` враћа ширину колоне, `GetColumnTitle` враћа назив табеле и још много других метода за манипулацију са табелама.

ReadFile објекти се користе за читање линија текста из фајла. Могућност читања у овом објекту поседује своје баферовање података што га чини бржим у односу на стандардну функцију за читање. ReadFile подржава различите типове формата фајлова. Неке од метода које се налазе у овом објекту су: `Open (filename, maxLineLength)` за отварање фајла за читање где се као улаз уносе име фајла и максимална дужина било које линије која ће се читати из фајла. `Read (void)` за читање података из отвореног фајла, ово је празна функција. `AtEOF (void)` проверава да ли је дошло до краја фајла, враћа два кода: `FE_OK` достигнут је крај фајла и престаје са читањем података и `FE_FAIL` није дошло до краја фајла и обавештава да има још података за читање. `Close(void)` за затварање фајла који се читао, `Jump (jumpto)` за прелазак (скок) на одређену локацију, `JumpToEnd (jumpto)` за прелазак (скок) на крај фајла, `Skipped (void)` за обустављање одређених команди, `Copy (void)` за копирање.

Selector објекат омогућава приступ функцијама за селектовање у FEMAP моделу. Користи се `feSelector` метода у FEMAP апликационом објекту за приступ Selector-у објекта. Неке од метода које припадају овом објекту су: `ClearAll (void)` ова метода брише све ентитете било ког типа, `Count (entityType, nSelected)` ова метода

враћа број ентитета датог типа који су тренутно селектовани, `SelectBox (void)` приказује дијалог за селекцију ентитета, `SelectPolygon (void)` приказује дијалог за селекцију полигона. `Add (entityType, ID)` додаје ентитет већ одабраним ентитетима, `GetSelected (entityType)` ова метода креира и враћа сет објеката који садрже ID-еве свих ентитета специфицираног типа који су тренутно селектовани.

Set објекат служи за чување листа ID-ова. Одређени Set објекат када се креира може бити додељен другим функцијама да врше неке операције на групи или више ентитета. Они могу бити сачувани и може им се приступити било када, из друге скрипте или програма. Неке од метода које припадају сет објекту су: `Add(id)` додаје ид у сет, `AddRange (startID, stopID, increment)` ова функција додаје домен одабраних ид-ева у сет, `AddSet (addID)` додаје све одабране ид-еве из другог сета у тренутни сет. `AddRule (addID, ruleID)` функција додаје у сет користећи нека од стандардних правила. (`FGD_Elem_byMatl` – бира елементе користећи ID материјала, `FGD_Elem_byProp` – бира елементи користеће ID својства, `FGD_Load_byNode` – бира силе које делују на одређени чвор,...).

3.4 Типови података

Промењљиве су места у меморији у којима подаци могу привремено да се чувају да би програм могао да их користи. Оне имају тип, име и вредност. Вредност промењљиве може да се мења за време извршавања програма, због чега је и добила назив – промењљива. Да би уопште могла да се употреби, промењљива најпре мора да се декларише : њен тип мора да се наведе и мора јој се дати име. Тип промењљиве дефинише дозвољени распон вредности које тај тип може да садржи, као и операције које могу да се обављају на датој промењљивој.

Типом података дефинисан је начин регистровања података у меморији, скуп могућих вредности тих података и скуп могућих акција над подацима.

Због различите дефиниције типова података у зависности од тога да ли је API програм написан у Basic-у, C++, или неком другом језику типови података су приказани у табели 3.4. Типови података који се користе су различити у зависности од језика који се користи, али морају одговарати формату типа података који задовољавају потребе API-ја.

Промењљиву декларишемо на следећи начин:

```
Dim naziv_promenljive As tip_podataka
```

Декларација промењљиве увек почиње наредбом `Dim`, након чега је обавезно да наведемо њен назив. Иза промењљиве се може навести и њен тип.

Табела 3.4. Врсте података

API definisan prema uputstvu	**Opis**	**Visual Basic 6**	**Visual Basic .NET**	**C++**
BOOL	True/ False	Boolean	Boolean	Unsigned Char
INT2	2-bitni celobrojni	Integer	Integer	short
INT4	4-bitni celobrojni	Long	Integer	long, int
REAL4	4-bitni realni	Single	Single	float
REAL8	8-bitni realni	Double	Double	double
STRING	string karaktera	String	String	char[..]

Један од типова података који се често користи је INT4. Приликом њеног декларисања мора се водити рачуна уколико користимо Visual Basic програмски језик. За декларисање 4-битног целобројног типа података у Visual Basic 6 користи се тип података Long док у Visual Basic .NET програмском језику Integer. Коришћењем типа података Long у програмском језику Visual Basic .NET појавиће се некомпактибилност типова података. Приликом коришћења Visual Basic .NET програмског језика податке типа STRING потребно је претходно иницијализовати, пре позивања у функцију. Пример овог је коришћење функције `feNotesGet` која преузима белешке које су написане у FEMAP моделу.

```
Dim beleska As String
beleska = ""
rc = App.feNotesGet(1,beleska)
```

На следећем примеру биће приказано креирање низа од 10 чланова који садржи 4-битне целобројне вредности:

```
Dim niz(10) As Long
niz(0) = 1
niz(1) = 2
```

Димензионисање квадратне матрице димензије (10,10) реалног типа података се може извршити на следећи начин:

```
Dim matrica(10,10) As Double
```

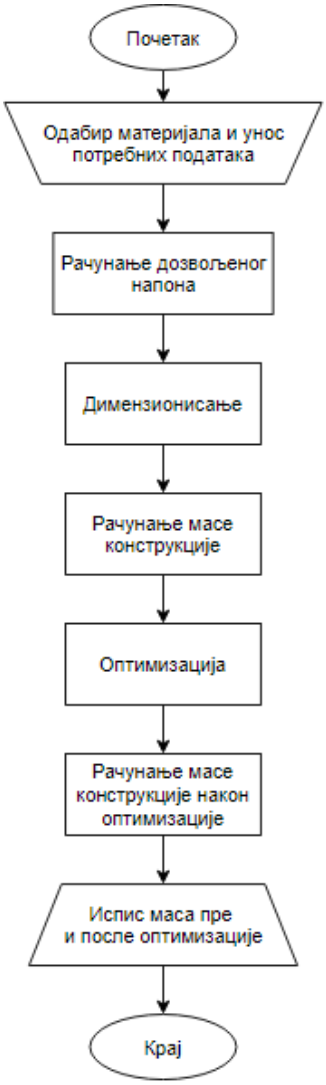
Логичке промењиве могу имати вредност тачно (True) или нетачно (False), а декларишемо их уз помоћ Boolean наредбе

```
Dim n as Boolean
n = True
```

4 Алгоритам за параметарску оптимизацију

Основна идеја алгоритма за параметарску оптимизацију решеткастих конструкција је приказана на слици 4.1. Алгоритам се односи на кораке које је потребно одрадити независно од начина рада, било то ручно или аутоматизовано уз коришћење API програмирања. Прво је потребно одабрати жељени материјал конструкције, након чега је потребно одредити дозвољени напон, на основу напона течења изабраног материјала и предложеног степена сигурности. Дозвољени напон се рачуна као количник напона течења материјала и степена сигурности (једначина 4.1).

$$\sigma_y = \frac{\sigma_{doz}}{S_s} \tag{4.1}$$



Слика 4.1 – Општи алгоритам за оптимизацију

Уз помоћ добијеног дозвољеног напона, одређене максималне силе и предложеног облика јединичног попречног пресека димензионишемо штапове у решеткастој конструкцији. Када се изврши димензионисање штапова у решеткастој конструкцији рачуна се њена маса пре оптимизације, након чега следи оптимизација и поновно рачунање масе оптимизоване решеткасте конструкције. На крају програм исписује масу пре и после оптимизације

Да би се извршило димензионисање штапова у разматраној решеткастој конструкцији, први корак је налажење максималне вредности сила које делују у штаповима (апсолутна вредност максималне силе). Штаповима је за почетак потребно задати јединични попречни пресек након чега је потребно покренути анализу и наћи расподелу аксијалних сила у штаповима разматране решеткасте конструкције. Након тога налазимо максималну вредност силе по апсолутној вредности због знака аксијалне силе односно карактера напрезања (притисак или затезање).

Димензионисање минималног попречног пресека штапа се врши коришћењем једначине 4.2.

$$A_{min} = \frac{F}{\sigma_{doz}} \tag{4.2}$$

Као репрезентатини пример облика попречног пресека у раду је узет кружни попречни пресек штапова при димензионисању. Када се пронађе минимални попречни пресек, рачуна се пречник за добијену вредност попречног пресека и затим се усваја прва већа вредност пречника. Пречник се рачуна на основу једначине 4.3.

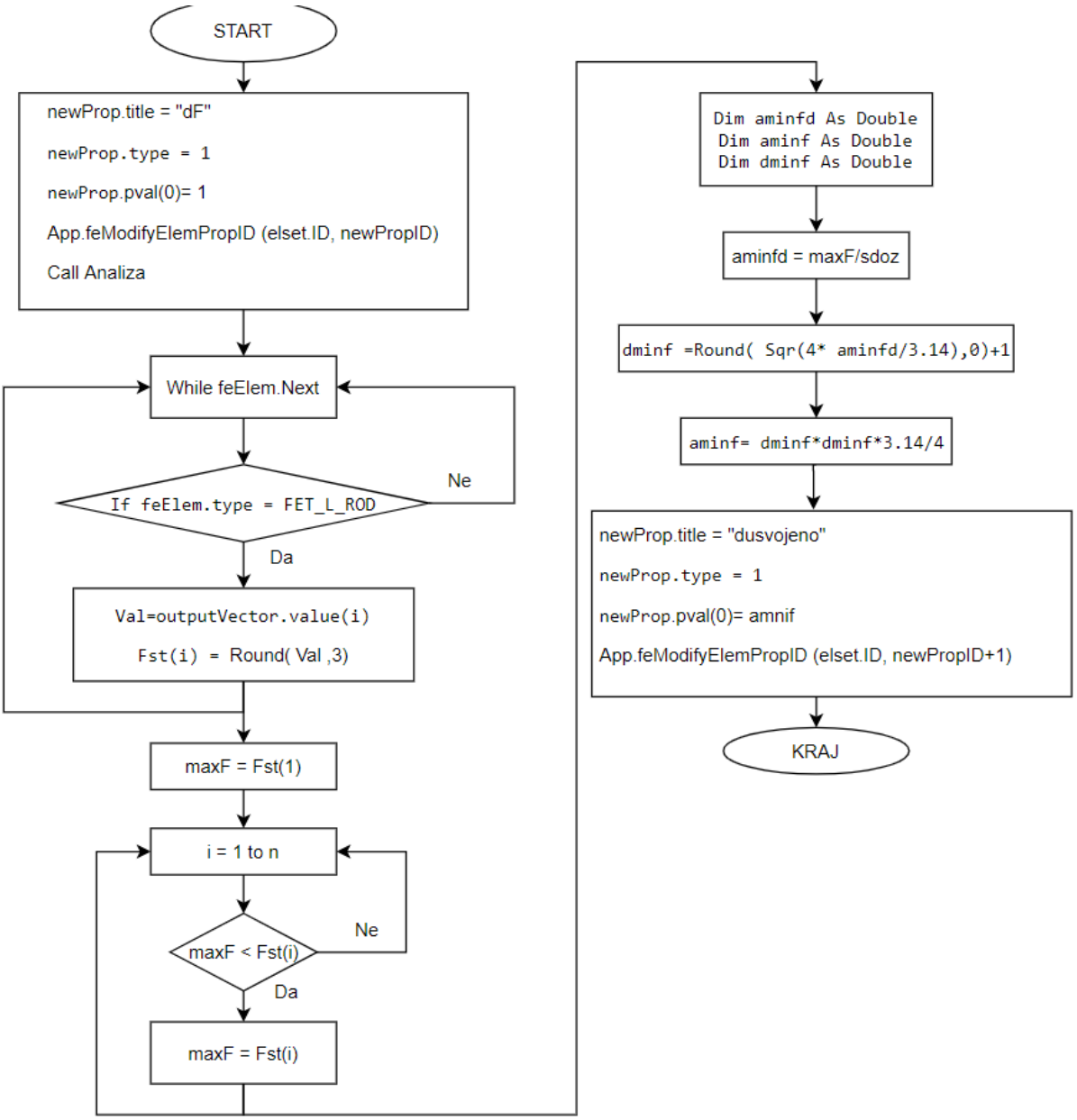
$$d = \sqrt{\frac{4A_{min}}{\pi}} \tag{4.3}$$

Усвојени пречник се користи за налажење површине попречног пресека, једначина 4.4.

$$A = \frac{d^2\pi}{4} \tag{4.4}$$

Усвајањем новог реалног попречног пресека поступак димензионисања је завршен. Када се димензионишу сви штапови, односно кад им се свима додели вредност израчунатог попречног пресека, рачуна се маса решеткасте конструкције пре оптимизације.

На слици 4.2 је приказан алгоритам за димензионисање уз помоћ API програмирња. Након преузимања сила које делују у штаповима, API програм налази највећу силу тако што се прва сила узима као највећа, након чега се остале упоређују са њом, ако је нека од њих већа она постаје највећа сила.

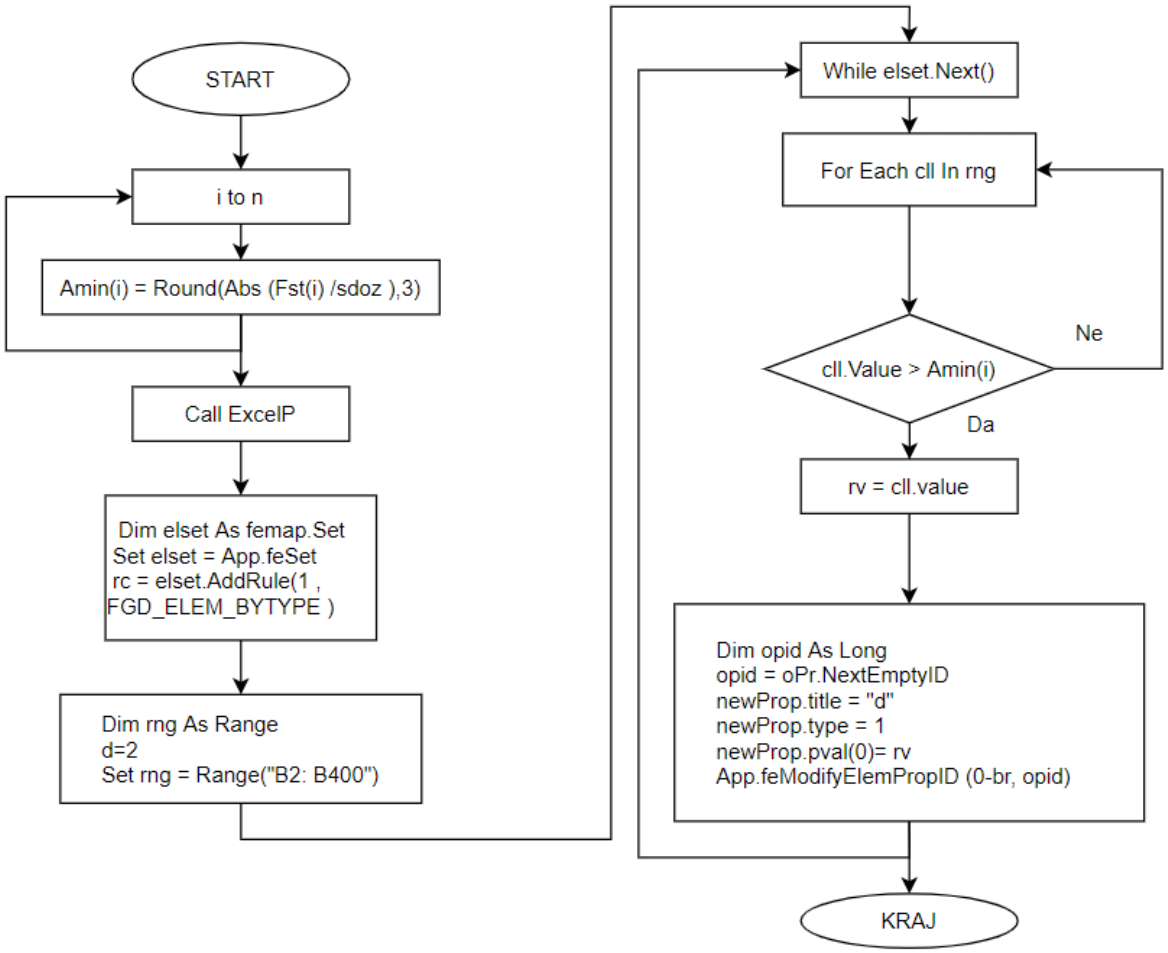


Слика 4.2 – Алгоритам за димензионисање штапова

Први корак у приказаном алгоритму на слици 4.3 је декларисање новог низа у коме се налазе вредности минималне површине попречног пресека за сваки елемент (штап) решеткасте конструкције. Минимална површина попречног пресека се рачуна као количник силе која делује на штап и срачунатог дозвољеног напона. API програм се повезује са табелом написаном у Excel-у, која садржи податке о предефинисаним површинама попречних пресека за одређене вредности пречника (за пречнике су узети цели бројеви, а могу бити и било које вредности по жељи корисника).

Креира се нови сет елемената у коме се налазе сви елементи штапа који саставни делови активног модела. `While` петљом се пролази кроз креирани сет елемената. Задаје

се нова променљива `cll` која ће представљати ћелију и преузимати вредности из задатог опсега. За сваку ћелију у табели унутар задатог опсега проверава се услов према коме вредност унутар ћелије треба бити већи од минималног попречног пресека штапа. Ако је услов испуњен програм враћа вредност унутар ћелије као нову вредност површине попречног пресека и прекида петљу, уколико није испуњен прелази на следећу. Након усвајања нове површине попречног пресека креира се и додељује штапу ново својство - Property (минимални попречни шресек штапа). Завршава се `While` петља и прелази се на следећи елемент, све док има елемената у сету.



Слика 4.3 – Алгоритам за оптимизацију решеткасте конструкције

Приликом отварања прозора API Programming који се налази у склопу софтверског пакета FEMAP, аутоматски ће се генерисати следећи код:

```
Sub Main

    Dim App As femap.model

    Set App = feFemap()
```

```
End Sub
```

Прва линија кода `Sub Main` представља почетак главног програма, док `End Sub` означава крај програма. Кодом између ове две линије дефинише се апликациони објекат који повезује API програм са тренутним активним моделом у FEMAP-у.

Прво што је потребно да програм одради је да обрише податке претходне анализе, па је за овај корак је написан је потпрограм који ће се више пута позивати у главном програму. Назив овог потпрограма је `DelOut`, у главном програму се позива на следећи начин: `Call DelOut`. Потпрограм се креира Sub и назив којим ће се касније користити за његово позивање.

```
Sub DelOut
```

Потребно је извршити димензионисање и повезати са тренутном сесијом (сетом), `oSet`, у којој ће се налазити резултати анализе и у коју ће се додавати сви резултати осталих анализа.

```
Dim oSet As femap.Set

Set oSet = App.feSet

oSet.AddAll( FT_OUT_CASE )
```

Након димензионисања и додавања резултата у сет, потребно је ресетовати наредни показивач да би се касније позивањем Next функције вратио на први унос.

```
oSet.Reset( )
```

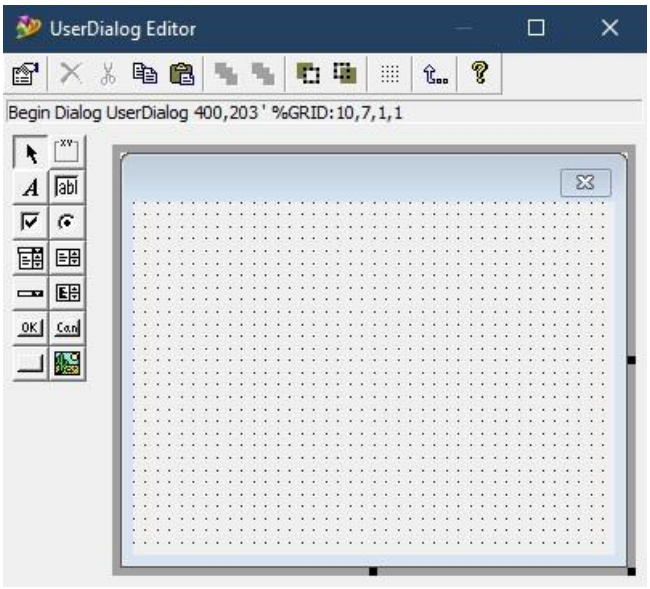
Креирана је while петља која служи за пролажење крос сет података у којој ће се све док постоји неки резултат брисати функцијом `feDelete` (`FT_OUT_CASE`, `oSet.CurrentID`), којој се унутар заграда додељује тип ентитета, у овом случају излаз анализе, и сет ентитета коришћењем идентификационог броја.

```
 While oSet.Next( )
      App.feDelete( FT_OUT_CASE, oSet.CurrentID)
 Wend
End Sub
```

Након брисања резултата претходних анализа, следећи корак је одабир материјала, за одабир се користи функција `mat.SelectID` која у зависности од објекта алата приказује нови прозор за селекцију.

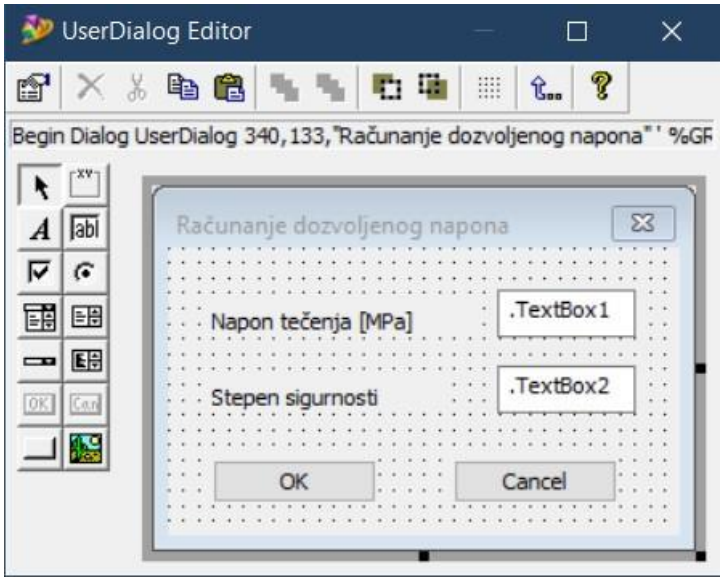
```
Dim mat As Matl

Set mat = App.feMatl

mat.SelectID( "Odabrati materijal")
```

Након одабира материјала креира се нови прозор за дијалог у коме ће се задати граница течења материјала у MPa и степен сигурности. У склопу програма за писање програма који се налази у FEMAP-у постоји могућност креирања дијалог прозора, (слика 4.4).



Слика 4.4 – Прозор едитора корисничких дијалога

У њему су додате два поља за текст, два дугмета: за потврђивање уноса и одустајање са додат одговарајућим текстом. Изглед готовог прозора је приказан на слици 4.5.



Слика 4.5 – Прозор едитора корисничких дијалога са додатим елементима

Унутар API програма потребно је исписати одговарајући код који се односи на тип и позиције елемената унутар дефинисаног прозора за дијалог.

```
    Begin Dialog UserDialog 340,133,"Računanje dozvoljenog
napona" ' %GRID:10,7,1,1
          TextBox 220,21,90,21,.TextBox1
          TextBox 220,56,90,21,.TextBox2
          text 30,28,180,14,"Napon tečenja [MPa]",.Text1
          text 30,63,160,14,"Stepen sigurnosti",.Text2
          OKButton 30,98,110,21
          CancelButton 190,98,110,21
     End Dialog
     Dim dlg As UserDialog
```

Потребно је додати услов у случају да се одустане од уноса потребних података. У случају одустајања програм ће ићи на линију OK која се налази на самом крају API програма и прекинуће извршавање програма.

```
     If Dialog(dlg) = 0 Then
          GoTo OK
     End If
```

Како би се искористили унети подаци потребно је декларисати промењиве и преузети вредност унетих података. Декларисане промењиве су:

```
Dim sigma As Double   ' напон течења
Dim ss As Double      ' степен сигурности
Dim sdoz As Double          ' дозвољени напон
```

Вредност се преузима из текст боксова као реални број типа Double.

```
sigma = Val(dlg.TextBox1 )
ss = Val(dlg.TextBox2)
```

За рачунање дозвољеног напона користи се једначина 4.1. У API програму:

```
sdoz = Round(sigma/ ss,3)
App.feAppMessage(0,"Dozvoljeni napon je: " + Str(sdoz) +
"MPa")
```

Резултат израчунатог дозвољеног напона се заокружује на три децимале и исписује на екрану.

Креира се сет у којем се налазе сви елементи типа штап. Прво је потребно декларисати сет у коме ће се налазити идентификациони бројеви штапова. Поред овога декларисан је и бројач n.

```
Dim elset As femap.Set
Set elset = App.feSet
Dim n As Long
rc = elset.Clear()
```

Штапови се убацују користећи функцију `AddRule` у којој је потребно унети тип ентитета и правило. Коришћено правило је `FGD_ELEM_BYTYPE` које сортира елементе по типу, ID број 1 означава елементе штапа.

```
rc = elset.AddRule(1 , FGD_ELEM_BYTYPE )
      n = elset.Count()
```

За потребе је креирања новог својства (Property), прво се декларише својство и потребно је декларисати ID број новог својства. За идентификациони број новог својства додељује се апсолутна вредност следећег слободног ID броја. Додељује се назив "dF", тип својства 1 за елемент типа штап, и додељује се вредност површине попречног пресека 1. Затим се додељује функцијом `.Put ID` број својства. Функција `.feModifyElemPropID` служи за додељивање новог својства елементима који се налазе у претходно креираном сету елемената.

```
Dim newProp As femap.Prop
Set newProp = App.feProp
Dim newPropID As Long
newPropID =Abs( newProp.NextEmptyID)
newProp.title = "dF"
newProp.type = 1
newProp.pval(0)= 1
newProp.Put(newPropID)
App.feModifyElemPropID (elset.ID, newPropID )
```

Следећи корак је позивање потпрограма Analiza. Декларише се нови објекат алата као `.AnalysisMgr`, декларише се нова промењљива aID за идентификациони број анализе.

```
Sub Analiza
      Dim Analiza As femap.AnalysisMgr
      Set Analiza = App.feAnalysisMgr
      Dim aID As Long
```

За анализу поља напона у решеткастој конструкцији извршава се статичка анализа у солверу NXNastran. Статичкој анализи се додељује одговарајући назив или назив по избору (Staticka analiza), затим се бира солвер, програм који обавља израчунавање и враћа резултате (NX Nastran). Потребно је декларисати тип анализе, `FAT_STATIC` за статичку анализу.

```
Analiza.title = "Staticka analiza"
Analiza.Solver =FAM_NX_NASTRAN
Analiza.AnalysisType = FAT_STATIC
```

За потребе анализе, потребно је преузети граничне услове - ограничења и оптерећења (силе) која делују на модел. У случају да нема ограничења или оптерећења API програм ће исписати поруку и прекинуће програм. Додељивање ограничења и оптерећења се врши додавањем сета анализи `BCSet (0)` за ограничења и `BCSet (2)` за оптерећења.

```
conset = App.Info_ActiveID(FT_BC_DIR)
If conset = 0 Then
     App.feAppMessageBox(0,"Nema ogranicenja.")
     Exit Sub
End If
Analiza.BCSet (0)=conset
ldset = App.Info_ActiveID (FT_LOAD_DIR)
If ldset =0 Then
     App.feAppMessageBox(0,     "Nema     aktivnog     seta
     opterecenja.")
     Exit Sub
End If
Analiza.BCSet (2) = ldset
```

За потребе одређивања шта ће бити резултат анализе, за приказ резултата анализе користи се функција `Output()` док се у заградама уписује захтев 15 за силе које делују у елементима, 16 напон који делује у елементима.

```
Analiza.Output(15) = -1
Analiza.Output(16) = -1
aID = Analiza.NextEmptyID
Analiza.Put(aID)
```

```
Analiza.Analyze(aID)
```

Након тога потребно је убацити `While` петљу која ће проверавати да ли су резултати учитани, па ако нису убациће паузу од 5 секунди. Када се заврши анализа исписује се порука да је анализа завршена.

```
While App.Info_Count(FT_OUT_CASE) <1
      Wait(5)
Wend
App.feAppMessage(FCM_NORMAL,"Analiza je zavrsena.")
```

Пре димензионисања штапова API програм преузима резултате анализе, како би одредио силе које делују у штаповима. Декларише се објекат излазног сета као и вектор. Излазном вектору се додаје ид број 3036 који одговара аксијалним силама које делују у штаповима.

```
Dim OSet As femap.OutputSet
Set OSet = App.feOutputSet

Dim outputVector As femap.Output
Set outputVector = App.feOutput

desiredOSetID=1
desiredOutputVectorID=3036

rc=OSet.Get(desiredOSetID)
outputVector.Get(desiredOutputVectorID)
```

Резултатима анализе се приступа функцијом `.Get()` где се преузимају вредности напона који делују у елементима.

Да би се резултат користио потребно га је доделити некој променљивој, низу axial. Креира се објекат алата `feElem` са којим се уз помоћ `While` петље пролази кроз елементе које задовољавају прописани услов, да је елемент штап, ако јесте бројач ai се повећава за 1, а у низу са одговарајућим индексом се памте вредности заокружене на три децимална броја.

```
Dim feElem As femap.Elem
Set feElem = App.feElem
Dim axial(1000) As Double
```

```
Dim ai As Long
ai =0
While feElem.Next
 If feElem.type = FET_L_ROD Then
      ai =ai + 1
      Val=outputVector.value(ai)
      axial(ai) = Round( Val ,3)
      msg = "Za stap " + Str$(feElem.ID) + " sila iznosi" +
      Str$( axial(ai) )
      rc = App.feAppMessage( FCM_NORMAL, msg )
 End If
Wend
```

Након преузимања вредности сила које делују у елементима, API програм налази силу највећег интензитета по апсолутној вредности. За налажење максималне силе прво се декларише нова променљива `maxF` која на почетку преузима вредност прве силе. Петљом `For` се пролази кроз низ са резултатима, па уколико је нека вредност у низу по апсолутној вредности већа од `maxF` она постаје `maxF`. Када се заврши петља `For` исписује се порука о највећој сили и њена вредност.

```
Dim maxF As Double
maxF=axial(1)
For i = 1 To ai
      If maxF < Abs(axial(i)) Then
           maxF = Abs(axial(i))
      End If
Next
      por="Najveca vrednost sile je " + Str$(maxF)
App.feAppMessage(FCM_NORMAL, por)
```

Након одређивања вредности максималне силе, програм налази минимални попречни пресек на основу максималне вредности силе и дефинисаног дозвољеног напона.

API програму прво је потребно декларисати променљиве за рачунање минималне површине попречног пресека за максималну вредност силе, aminfd, пречник и површину попречног пресека.

```
Dim aminfd As Double

Dim aminf As Double

Dim dminf As Double

aminfd = maxF/sdoz   `рачунање минималног попречног пресека за највећу силу
```

За усвајање пречника попречног пресека потребно је заокружити добијену вредност из једначине 4.3 на цели број (одсецају се децимале) и додати 1.

```
dminf =Round( Sqr(4* aminfd/3.14),0)+1
```

Када се усвоји пречник рачуна се површина попречног пресека и исписује се њена вредност на екрану.

```
aminf= dminf*dminf*3.14/4

     por = "Minimalna povrsina poprecnog preseka na osnovu
     maksimalne sile "+ Str$(aminf)

App.feAppMessage(FCM_NORMAL, por)
```

Декларише се нови празан низ који се редекларише како би имао вредност укупног броја штапова. У овом низу памте се резултати минималног попречног пресека за све штапове, на претходно поменут начин.

```
Dim Amin() As Double

ReDim Amin(n) As Double

For i = 1 To n

Amin(i) = Round(Abs(axial(i))/ sdoz,3)

Next
```

Након усвајања површине попречног пресека креира се ново својство (површина попречног пресека) dusvojeno са новом вредношћу површине попречног пресека.

```
newProp.title = "dusvojeno"

newProp.type = 1

newProp.matlID =CLng(mat.ID)

newProp.pval(0)= aminf

newProp.Put(newPropID+1)

App.feModifyElemPropID (elset.ID, newPropID+1)
```

Након што је креирано и додељено ново својство са измењеном вредношћу површине попречног пресека, API програм позива потпрограме за брисање резултата и

анализу. Да би се одредила маса конструкције након димензионисања потребно је задати убрзање земљине теже позивањем потпрограма `Bload`.

```
Sub Bload
Dim gr As femap.LoadSet    'декларисање оптерећења
Set gr = App.feLoadSet
Dim grID As Long
grID =Abs( gr.ID)
```

Да би и сила сопствене тежине била узета у обзир у прорачуну задаје се одговарајућа вредност убрзања земљине теже у правцу вертикалне осе решеткасте конструкције.

```
gr.BodyAccelOn = True
gr.BodyAccel(1) = -9810
gr.Put(grID)
End Sub
```

Када се сила земљине теже узме у обзир позива се други API програм за рачунање масе, коме се прослеђује у виду стринга команда која садржи вредност сета у коме се налазе штапови.

```
MacroRun CurDir$() & "\Masa.BAS",Str(elset.ID)
```

Програм за рачунање масе преузима прослеђену команду и врши конверзију у други тип података, стринг у целобројни тип података.

```
elsetID = CInt(Command$)
```

Функција `feMeasureMeshMassProp` се користи за проналажење укупне масе конструкције.

```
App.feMeasureMeshMassProp(elsetID, 0, False, False, Len1,
area, volume, structMass, nonStruct, TotalMass, structCG,
nonstructCG, totalCG, inertia, intertiaCG)
```

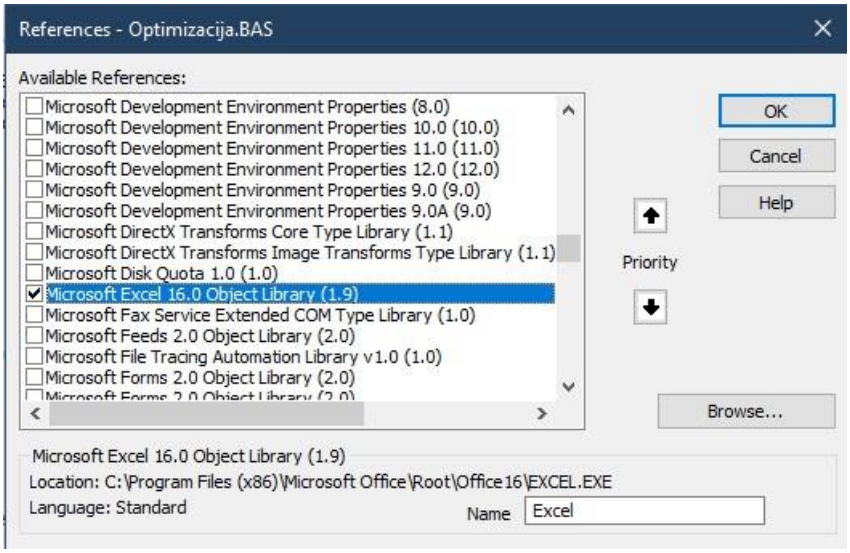
Израчуната маса конструкције након димензионисања се штампа као порука. Тренутна вредност масе је у тонама, а да би се маса конструкције добила у килограмима потребно је резултат помножити са 1000.

```
por = Str(Round(TotalMass,6)*1000)+"kg"
App.feAppMessage (FCM_NORMAL , por)
```

Комплетан код за рачунање масе дат је у прилогу 2.

API програм даље позива потпрограм за повезивање са Excel датотеком у којој се налази одговарајућа табела са предефинисаним пречницима и површинама попречних

пресека. Да би функције Excel-а могле да се користе потребно је додати референцу. Референца се додаје тако што активира десни клик миша на делу у којем се уноси код, након чега се појављује прозор приказан на слици 4.6.



Слика 4.6 – Прозор за референце

У овом прозору потребно је наћи и обележити библиотеку за објекте Excel-а. У супротном пријављиваће грешку за непостојећи тип објеката

```
Sub ExcelP
```

Декларише се апликативан објекат за приступ Excel-у.

```
Dim appExcel As Excel.Application
Set appExcel =  New Excel.Application
```

Декларише се објекат за приступ Excel датотеци, као и путања до одговарајуће датотеке.

```
Dim wrkBook As Excel.Workbook
Set wrkBook = appExcel.Workbooks.Open(CurDir$() & "\femap_pod.xlsx")
```

Декларише се објекат за приступ радним листовима изабране датотеке написане у Excel-у.

```
Dim wrkSheet As Excel.Worksheet
Set wrkSheet = wrkBook.Worksheets(1)
```

End Sub

Декларише се нови низ за површину попречног пресека као и ново својство.

```
Dim Ar() As Double
ReDim Ar(n) As Double
```

```
Dim oPr As femap.Prop
Set oPr = App.feProp
```

Декларисање променљивих за пречник и индекс елемената у креираном низу.

```
Dim d As Integer
Dim br As Integer
br = 0
```

`While` петља за пролаз кроз елементе који се налазе у сету у којем се налазе штапови. Унутар петље се налази бројач, позива се потпрограм `PPPR(Amin(br),Ar(br),d)` којем се прослеђују вредности сила које делују на штапове, претходно декларисани празни низ и променљива за пречнике. Потпрограм враћа резултате и они се уписују у ново својство. За сваки елемент типа штап који се налази у моделу креираће се ново својство које има назив d + вредност пречника за одговарајући попречни пресек.

```
While elset.Next()
      br = br+1
      Call PPPR(Amin(br),Ar(br),d)
      Dim opid As Long
      opid = oPr.NextEmptyID
      oPr.title = "d" +Str(d)
      oPr.type = 1
      oPr.matlID =CLng(mat.ID)
      oPr.pval(0)= Ar(br)
      oPr.Put( opid)
      App.feModifyElemPropID (0-br, opid)
Wend
```

Потпрогам за усвајање површине попречног пресека на основу силе која делује на штап ради следеће:

```
Sub  PPPR(v As Double, rv As Double, d As Integer)
Dim rng As Range
d=2
```

Декларише се опсег у коме се налазе подаци о површини попречног пресека, од ћелије B2 до B400 у Excel датотеци.

```
Set rng = Range("B2: B400")
```

`For` петља која пролази кроз све ћелије које се налазе у декларисаном опсегу проверава услов ако је вредност у тренутној ћелији већи од вредности добијене од стране главног програма усвојена вредност попречног пресека rv преузеће вредност те ћелије и изаћи из петље, ако не прелази на следећу вредност.

```
For Each cll In rng
      d=d+1
      If cll.value > v Then
           rv=cll.value
           Exit For
      End If
 Next
End Sub
```

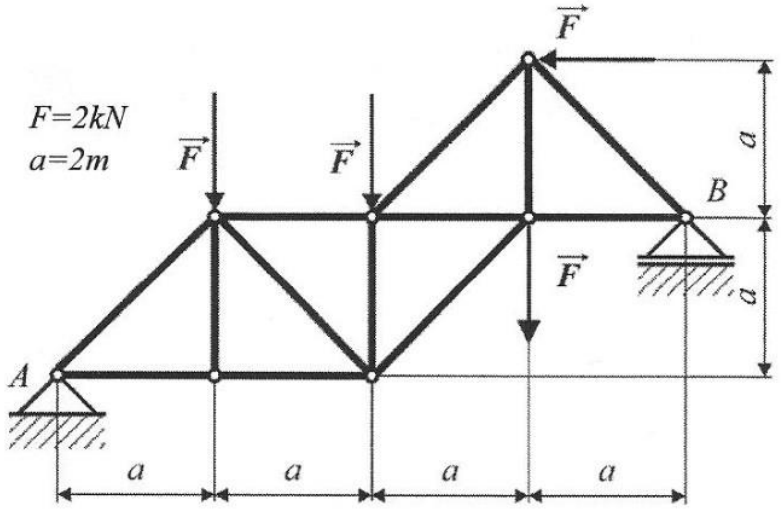
Затим API програм позива други API програм, за рачунање масе конструкције и исписује вредност масе конструкције након оптимизације; покреће анализу како би се касније приказали резулати напона у решеткастој конструкцији који морају бити мањи од вредности дозвољеног напона дефинисаног на основу напона течења материјала и изабраног степена сигурности.

Када се API програм сачува како би се касније позивао потребно је додати га у одељку за кориснички креиране алате. Додаје се тако што се из траке са алатима бира опција ***Custom Tools*** и из падајућег менија ***Add Tools***..., затим се бира одговарајући API програм у овом случају развијени програм ***Optimizacija*** за параметарску оптимизацију решеткастих конструкција.

Комплетан код API програма за параметарску оптимизацију решеткастих конструкција кружног попречног пресека дат је у прилогу 1.

5 Параметарска оптимизација изабране решеткасте конструкције

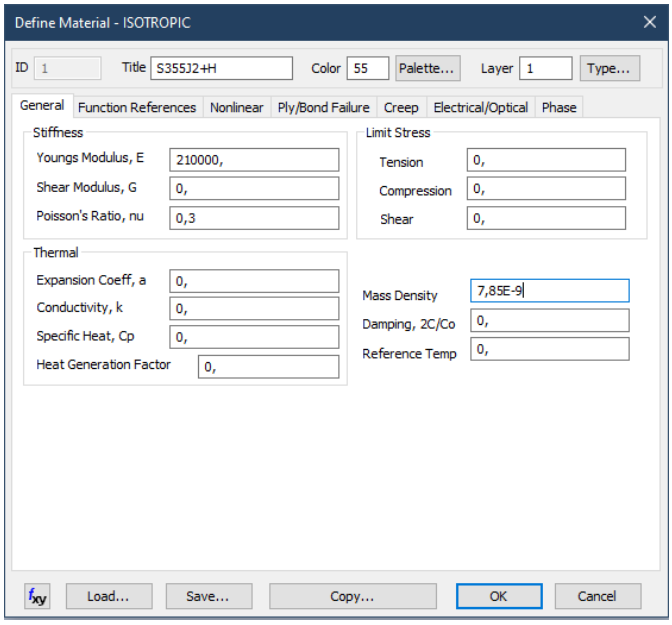
Поступак параметарске оптимизације решеткасте конструкције биће илустрован на примеру решеткастог носача приказаног на слици 5.1. Циљ је да на основу вредности изабраног напона течења и степена сигурности (σ_y = 355MPa; S = 1.33) као крајњи излаз добијемо оптимизовану решеткасту конструкцију са штаповима кружног попречног пресека најмање могуће масе, која у потпуности задовољава прописане критеријуме. Материјал од кога је израђен овај решеткасти носач је челик S355J2+N који има следеће материјалне карактеристике: модул еластичности E = 2.1 10^5 MPa, Поасонов коефицијент ν = 0.3 и густину материјала ρ = 7850 kg/m^3.



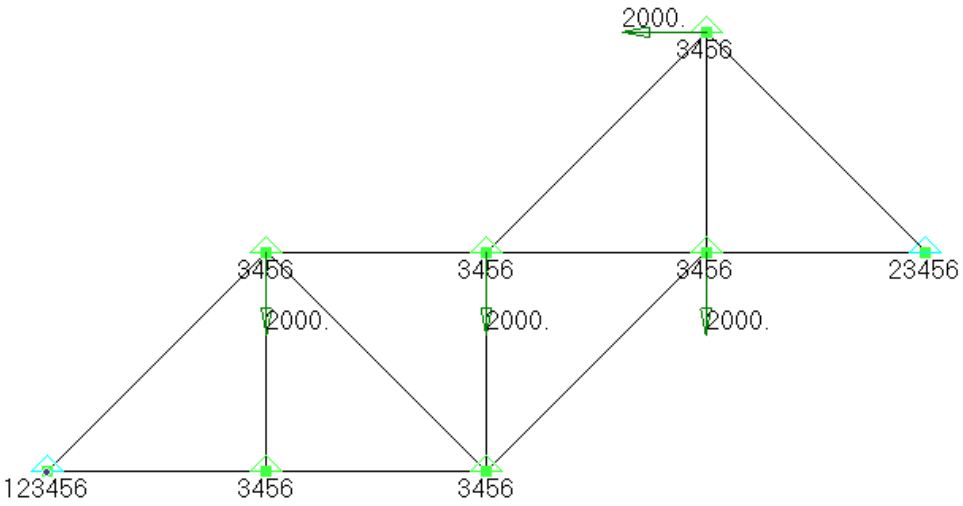
Слика 5.1 – Решеткасти носач оптерећен силама

Прво је потребно дефинисати катактеристике материјала избором команди ***Model/Material*** или их креирати из стабла. Након отварања прозора приказаног на слици 5.2 потребно је унети модул еластичности у MPa, Поасонов коефицијент и густину материјала у t/mm^3.

Пре креирања елемената потребно је дефинисати нови Property избором команди Model /Property и изабрати да је елемент штапа, и задати јединични попречни пресек. На слици 5.3 је приказан модел коначних елемената разматране раванске решеткасте конструкције са задатим ограничењима и оптерећењима у складу са поставком задатка приказаном на слици 5.1.

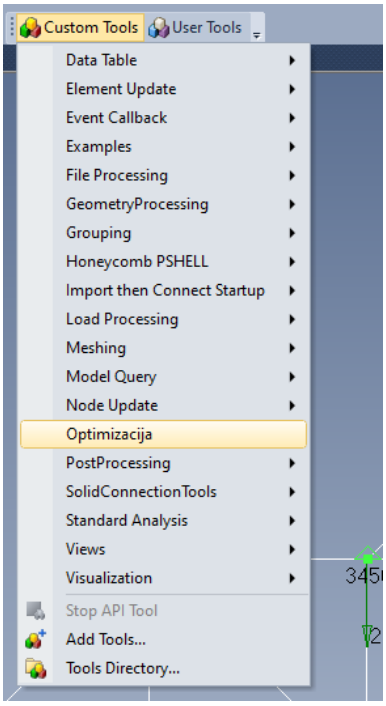


Слика 5.2 – Дефинисање материјала у FEMAP-у



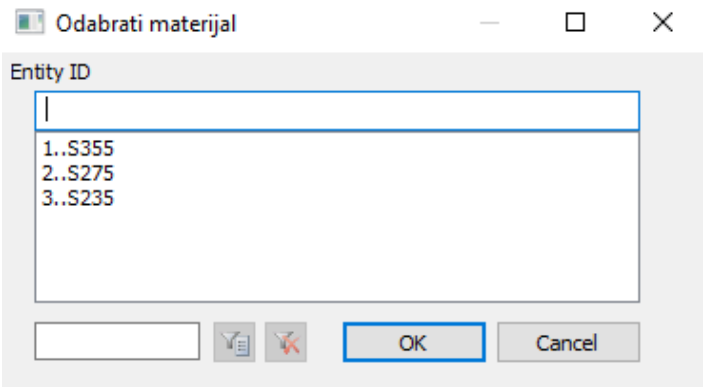
Слика 5.3 – Решеткасти носач оптерећен силама (FEMAP радно окружење)

Након креирања МКЕ модела потребно је позвати развијени API програм за параметарску оптимизацију. Позивање API програма се врши на следећи начин: потребно је изабрати траку са алатима ***Custom Tools***, након чега ће се појавити падајући мени у коме је потребно изабрати претходно додат алат ***Optimizacija***, (слика 5.4).



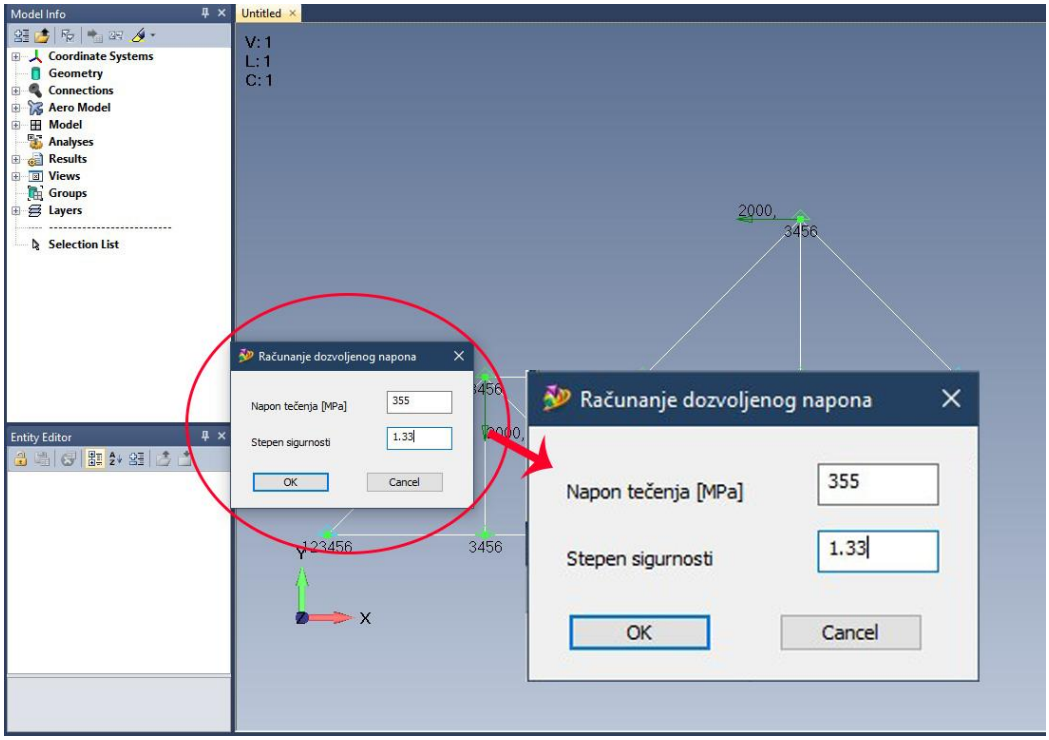
Слика 5.4 – Позивање API програма за оптимизацију

Када се покрене развијени API програм потребно је одабрати материјал. Уколико постоји више креираних материјала или ако је потребно анализирати исту решеткасту конструкцију израђену од различитих материјала појавиће се прозор са предефинисаним материјалнима, приказан на слици 5.5. У овом нашем примеру примеру бирамо S355 у складу са поставком задатка.



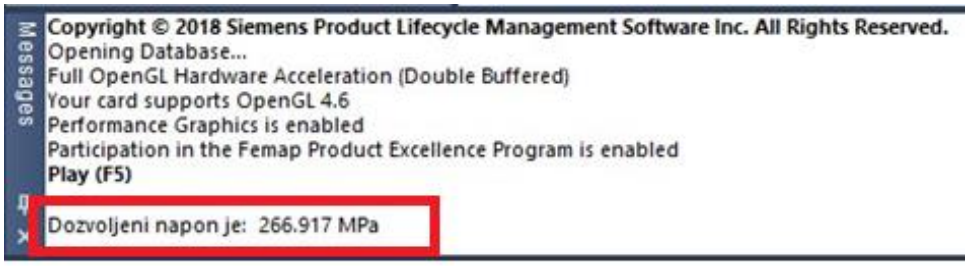
Слика 5.5 – Одабир материјала

Када се одабере материјал потребно је унети напон течења и степен сигурности, за материјал S355. Напон течења износи σy = 355 MPa, док задати степен сигурнонсти износи 1.33, који нам даље служе за израчунавање дозвољеног напона. Унос ових података је приказан на слици 5.6.



Слика 5.6 – Унос података у API програм за рачунање дозвољеног напона

Након уноса података за напон течења материјала и прописаног степена сигурности API програм рачуна дозвољени напон и вредност исписује у делу за поруке у FEMAP-у (слика 5.7).



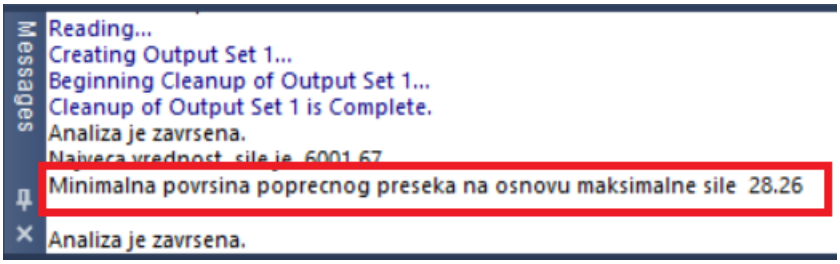
Слика 5.7 – Исписивање израчунатог дозвољеног напона

Када се израчуна дозвољени напон API програм покреће МКЕ анализу. За анализу користимо солвер NX Nastran који је саставни део програмског пакета FEMAP.

Резултат анализе је поље аксијалних сила у штаповима. На основу резултата и приказаног поља сила у штаповима API програм проналази максималну вредност силе која делује у разматраној решеткастој конструкцији. Због карактера напрезања (притисак или затезање) потребно је пронаћи максималну вредност силе по апсолутној вредности и након тога та вредност максималне силе се исписује у простору за поруке у FEMAP-у (слика 5.8).

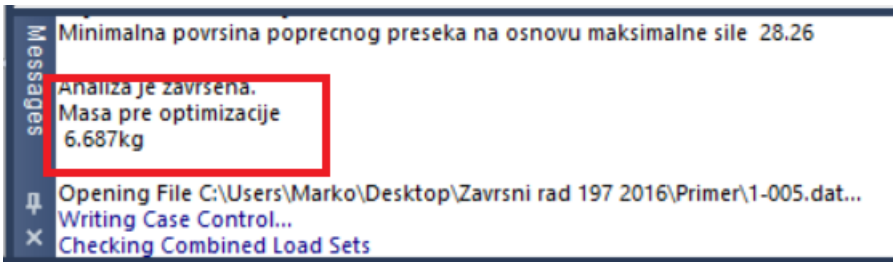
Слика 5.8 – Исписивање највеће вредности силе

На основу израчунате максималне вредности силе у решеткастој конструкцији потребно је извршити димензионисање, односно потребно је израчунати минималну површину попречног пресека и записати је у простору за поруке у FEMAP-у (слика 5.9).



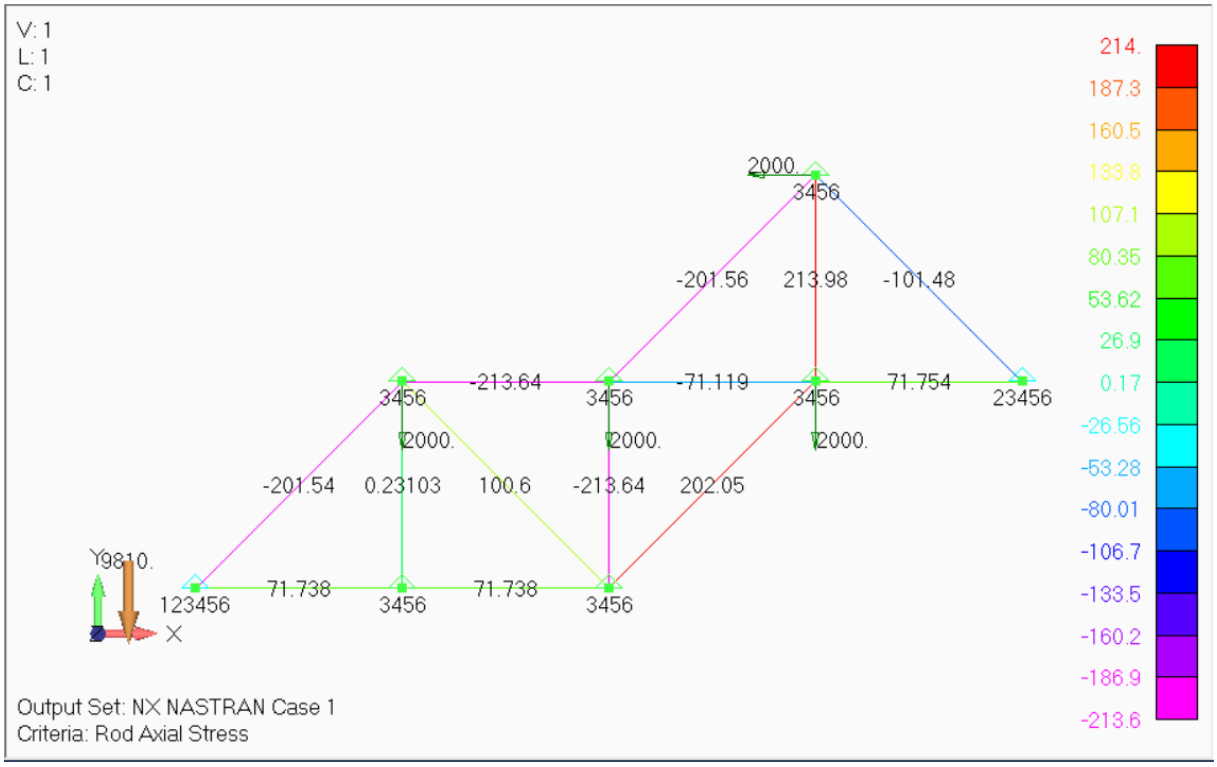
Слика 5.9 – Исписивање минималне површине попречног пресека

Када се пронађе усвојени минимални попречни пресек рачуна се маса почетне конструкције где сви штапови решеткасте конструкције имају исти попречни пресек који смо претходно одредили. За рачунање масе API програм позива функцију која задаје убрзање земљине теже. Повезује се са Excel табелом приказаној на слици 5.8, која садржи податке о пречнику у mm и површини предефинисаних попречних пресека у mm^2 и рачуна масу решеткасте конструкције у којој сви штапови имају исти попречни пресек, који смо у претходном кораку одредили. Након тога у простору за поруке у FEMAP-у исписује се почетна вредност масе конструкције (маса пре оптимизације), слика 5.10.



Слика 5.10 – Исписивање масе пре оптимизације

Као контрола да су све вредности напона у штаповима мање од претходно дефинисаног дозвољеног напона (потврда да је димензионисање добро урађено) приказујемо поље напона у решеткастој конструкцији (слика 5.11).



Слика 5.11 – Поље напона у решеткастој конструкцији након димензионисања

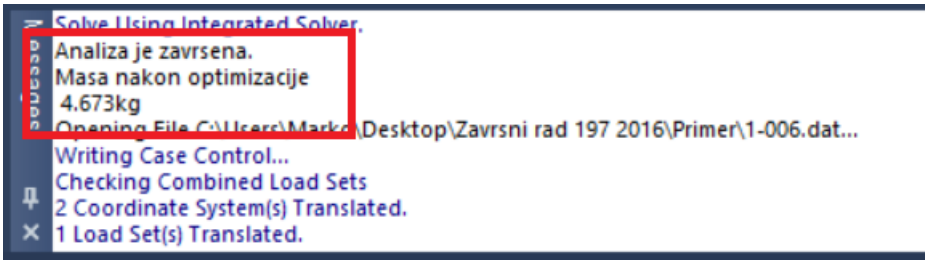
Након завршеног поступка димензионисања API програм приступа процесу оптимизације решеткасте конструкције тако што мање оптерећеним штаповима додељује најмању могућу вредност попречног пресека, а да претходно дефинисани услови буду испуњени (напони у штаповима мањи од дозвољеног напона). Неоптерећеним штаповима се додељује минимална вредност попречног пресека предефинисана у Excel табели (слика 5.12).

A1

	A	B
1	Diametar	Area
2	3	7,068583471
3	4	12,56637061
4	5	19,63495408
5	6	28,27433388
6	7	38,48451001
7	8	50,26548246
8	9	63,61725124
9	10	78,53981634
10	11	95,03317777
11	12	113,0973355
12	13	132,7322896
13	14	153,93804
14	15	176,7145868
15	16	201,0619298
16	17	226,9800692
17	18	254,4690049

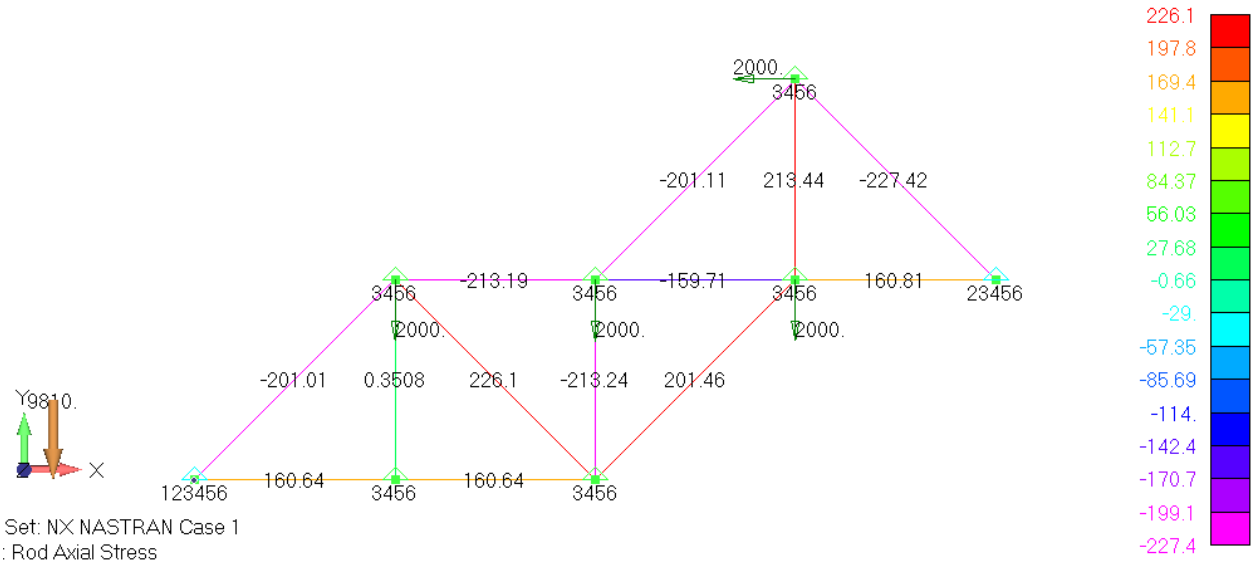
Слика 5.12 – Excel табела са предефинисаним кружним попречним пресецима

Сваком штапу се додељује нови property и API програм поново покреће анализу и исписује масу решеткасте конструкције после оптимизације у простору за поруке у FEMAP-у (слика 5.13).



Слика 5.13 – Исписивање израчунатог дозвољеног напона

Као контрола да су све вредности напона у штаповима мање од претходно дефинисаног дозвољеног напона (потврда да је процес параметарске оптимизације добро урађен) приказујемо поље напона у решеткастој конструкцији након оптимизације (слика 5.14).



Слика 5.14 – Поље напона у решеткастој конструкцији након оптимизације

Након што смо утврдили да су све вредности напона у штаповима мање од дозвољеног напона, дефинисаног на основу жељеног критеријума, као резултат добро извршене параметарске оптимизације решеткасте конструкције у простору за поруке у FEMAP-у исписане су вредности масе конструкције пре и после оптимизације.

6 Закључак

Решеткасте конструкције су веома погодне са становишта утрошка материјала уз високо искоришћење конструкције, имају мању масу у односу на пуне конструкције, омогућавају премошћавање великих распона. Због својих предности веома су погодне за примену у зградарству и мостоградњи. За потребе анализе оптерећења решеткастих конструкција потребно је познавати физичка и механичка својства материјала од којих су израђени и силе које делују на разматрану решеткасту конструкцију.

Метода коначних елемената као нумеричка метода се успешно користи за решавање решеткастих конструкција и пружа много више могућности са аспекта брзине и тачности у односу на „традиционалне“ методе за решавање решеткастих конструкција. Одређивањем сила у штаповима решеткасте конструкције за одговарајући материјал и са прописаним степеном сигурности могуће је одредити минимални попречни пресек штапова односно могуће је извршити њихово димензионисање.

Овако димензионисану решеткасту конструкцију је могуће даље оптимизовати и за ту потребу написан је API програм за параметарску оптимизацију који за крајњи излаз даје решеткасту конструкцију најмање могуће масе која задовољава све унапред прописане и постављене критеријуме. За креирање МКЕ модела решеткасте конструкције и приказ резултата коришћен је софтверски пакет FEMAP, док је за МКЕ анализу решеткасте конструкције коришћен софтвер NX Nastran.

У раду је приказан алгоритам оптимизације решеткасте конструкције преточен у API програм у коме је извршена оптимизација раванске решеткасте конструкције за кружни попречни пресек у коме се мења само један параметар (димензија), мења се пречник кружног попречног пресека. Уз мање измене могуће је извршити адаптацију за било који други облик попречног пресека у коме ће се мењати само једна димензија, на пример квадратни попречни пресек у коме би се мењала само страница квадрата. Такође постоји и могућност додавања опције за повезивања са новом табелом која ће садржати податке са материјалима различитих врста.

Коришћење API програмирања нам омогућује да на једноставан и ефективан начин проширимо могућности софтверског пакета FEMAP. Највећа предност API програмирања је аутоматизација неког процеса. Уместо човека, рачунар ће одратити посао и са тим ће се смањити и укупно време које је потребно за извршавање неког задатка. Може поједноставити или уклонити редудантне кораке неког процеса. API програмирање у склопу FEMAP-а нам омогућује брзо креирање или измену ентитета модела као што су чворови, елементи својства и слично. Омогућује повезивање са Excel-ом за брзу размену података између два програма као што је учитавање података из радних листова Excel-а или уписивање нових података у њих. Поред Excel-а могуће је повезивање и са другим програмима као што је Word, чиме можемо да креирамо извештаје. Коришћењем API програмирања можемо осигурати да се у неким програмима

комбинације изведу на исти начин више пута чиме је могуће смањити постојање грешака које би могле настати услед уноса података.

Једном написан API програм могуће је веома брзо и лако адаптирати за решавање сличних проблема. Поред тога постоји могућност и повезивања већ постојећих API програма са неким другим програмима и апликацијама. Све ово испред наведено нас доводи до закључка да коришћењем API програмирања можемо знатно повећати ефикасност и брзину у раду.

7 Литература

[1] Којић М, Мићуновић М: Статика, Научна књига, Београд, 1978.

[2] Којић М., Славковић Р., Живковић М., Грујовић Н.: Метод коначних елемената I, Машински факултет, Крагујевац, 1998.

[3] https://www.plm.automation.siemens.com/global/en/products/simcenter/femap.html, преузето септембар 2019.

[4] http://iprod.masfak.ni.ac.rs/resources/project_results/wp_4_3/UNIKG_-_MOTS.pdf, преузето септембар 2019.

[5] Ahttps://www.grf.bg.ac.rs/p/learning/p9_2017_2018_1513422827972.pdf, преузето септембар 2019.

[6] https://sr.wikipedia.org/sr-el/%D0%94%D0%B0%D1%82%D0%BE%D1%82%D0%B5%D0%BA%D0%B0:20090719-most-svilajnac-atipiks.jpg, преузето септембар 2019.

[7] https://cdn.cranemarket.com/images/listings/saez-slh-110-6-ton-hydraulic-luffing-jib-tower-crane-91728.jpg, преузето април 2020.

[8] https://docs.microsoft.com/en-us/cpp/mfc/ole-background?view=vs-2019, преузето септембар 2019.

ПРИЛОГ 1: API програм за параметарску оптимизацију

```
'#Reference                              {00020813-0000-0000-C000-
000000000046}#1.9#0#C:\Program      Files      (x86)\Microsoft
Office\Root\Office16\EXCEL.EXE#Microsoft   Excel   16.0   Object
Library#Excel



    Sub Main
        Dim App As femap.model
        Set App = feFemap()
        ' pozivanje potprograma za brisanje prethodnih analiza
     Call DelOut
    ' odabir materijala
     Dim mat As Matl
     Set mat = App.feMatl
      mat.SelectID( "Odabrati materijal")

     ' deklarisanje promenljivih
     Dim sigma As Double
     Dim ss As Double
     Dim sdoz As Double
     'kreiranje dijaloga za unos napona i stepena sigurnosti
     Begin  Dialog  UserDialog  340,133,"Računanje  dozvoljenog
napona" ' %GRID:10,7,1,1
          TextBox 220,21,90,21,.TextBox1
          TextBox 220,56,90,21,.TextBox2
          text 30,28,180,14,"Napon tečenja [MPa]",.Text1
          text 30,63,160,14,"Stepen sigurnosti",.Text2
          OKButton 30,98,110,21
          CancelButton 190,98,110,21
     End Dialog
```

```
     Dim dlg As UserDialog
     If Dialog(dlg) = 0 Then
          GoTo OK
     End If
     'zapisivanje promenljivih iz dijaloga
     sigma = Val(dlg.TextBox1 )
     ss = Val(dlg.TextBox2)
     ' racunanje i ispis dozvoljenog napona
     sdoz = Round(sigma/ ss,3)
     App.feAppMessage(0,"Dozvoljeni napon je: " + Str(sdoz) +
"MPa")
     Dim n As Long
        Dim elset As femap.Set
        Set elset = App.feSet
        Dim msg As String
        rc = elset.Clear()
        rc = elset.AddRule(1 , FGD_ELEM_BYTYPE )
     n = elset.Count()
     Dim newProp As femap.Prop
     Set newProp = App.feProp
     ' deklarisanje i inicijalizacije grupe za izolaciju ID
stapova
     Dim newGroup As femap.Group
     Set newGroup = App.feGroup
     Dim newGroupID As Long
     newGroupID = newGroup.NextEmptyID
     newGroup.SetAdd(FT_ELEM,elset.ID)
     newGroup.title = "Stapovi"
     newGroup.Put(newGroupID)
       ' kreiranje novog property-a i zadavanje elementima
          Dim newPropID As Long
```

```
          newPropID =Abs( newProp.NextEmptyID)
          newProp.title = "dF"
          newProp.type = 1
          newProp.pval(0)= 1
          newProp.Put(newPropID)
          App.feModifyElemPropID (elset.ID, newPropID)
          Dim upID As Long
          upID = newPropID
          App.feModifyElemPropID( newGroupID, upID)
          Call Analiza
          ' SILA KOJA DELUJE U STAPOVIMA
        ' dimenzionisanje sile i minimalnog poprecnog preseka za
stapove
        ' preuzeti vrednost sile iz prethodne analize
     Dim ouSetID As Long
        Dim ouVec As femap.Output
        Set ouVec = App.feOutput
        Dim feView As femap.View
        Set feView = App.feView
        Dim viewID As Long
        Dim axial(1000) As Double
          Dim OSet As femap.OutputSet
    Set OSet = App.feOutputSet
        Dim outputVector As femap.Output
        Set outputVector = App.feOutput
        desiredOSetID=1
        desiredOutputVectorID=3036
        rc=OSet.Get(desiredOSetID)
        outputVector.Get(desiredOutputVectorID)
        Dim feElem As femap.Elem
        Set feElem = App.feElem
```

```
      Dim ai As Long
      ai =0
         While feElem.Next
             If feElem.type = FET_L_ROD Then
                ai =ai + 1
                Val=outputVector.value(ai)
                 axial(ai) = Round( Val ,3)
                 msg = "Za stap " + Str$(feElem.ID) + " sila
iznosi" + Str$( axial(ai) )
                 rc = App.feAppMessage( FCM_NORMAL, msg )
             End If
         Wend
     ' NALAZENJE NAJVECE SILE KOJA DELUJE U STAPOVIMA
      Dim maxF As Double
      maxF=axial(1)

      For i = 1 To n
           If maxF < Abs(axial(i)) Then
                     maxF = Abs(axial(i))
           End If
      Next
           por="Najveca vrednost sile je " + Str$(maxF)
      App.feAppMessage(FCM_NORMAL, por)
      ' NALAZENJE MINIMALNOG POPRECNOG PRESEKA ZA NAJVECU
AKSIJALNU SILU
      Dim aminfd As Double
      Dim aminf As Double
      Dim dminf As Double
      aminfd = maxF/sdoz
      dminf =Round( Sqr(4* aminfd/3.14),0)+1
      aminf= dminf*dminf*3.14/4
```

```
     por = "Minimalna povrsina poprecnog preseka na osnovu
maksimalne sile "+ Str$(aminf)
     App.feAppMessage(FCM_NORMAL, por)
     ' RACUNANJE MINIMALNOG POPRECNOG PRESEKA
     Dim Amin() As Double ' dimenzionisanje niza za poprecni
presek
     ReDim Amin(n) As Double
     For i = 1 To n
     Amin(i) = Round(Abs(axial(i))/ sdoz,3)
     'por = "Minimalni povrsina poprecni presek za " + Str$(i)
+ " je " + Str$(Amin(i))
     'App.feAppMessage(FCM_NORMAL, por)
     Next
    ' PRAVLJENJE NOVOG PROPERTY-a NAKON DIMENZIONISANJA STAPA
          newProp.title = "dusvojeno"
          newProp.type = 1
    ' dodeljivanje ID materijala odabranog materijala
          newProp.matlID =CLng(mat.ID)
    ' dodeljivanje povrsine poprecnog preseka
          newProp.pval(0)= aminf
          newProp.Put(newPropID+1)
          App.feModifyElemPropID (elset.ID, newPropID+1)
     Call DelOut
     ' NOVA ANALIZA
     Call Analiza
     ' pozivanje potprograma za zadavanje ubrzanja zemljine teze
      Call BLoad
     ' racunanje mase
     App.feAppMessage(FCM_NORMAL, "Masa pre optimizacije")
     MacroRun CurDir$() & "\Masa.BAS",Str(elset.ID)
```

```
    Call ExcelP
    ' nova grupa za svaki element
     Dim Ar() As Double
     ReDim Ar(n) As Double
     Dim oPr As femap.Prop
     Set oPr = App.feProp
     Dim d As Integer
     Dim br As Integer
     br = 0
    While elset.Next()
          br = br+1
          Call PPPR(Amin(br),Ar(br),d)
          Dim opid As Long
          opid = oPr.NextEmptyID
          oPr.title = "d" +Str(d)
          oPr.type = 1
          oPr.matlID =CLng(mat.ID)
          oPr.pval(0)= Ar(br) ' dodeljivanje povrsine poprecnog
preseka
          oPr.Put( opid)
          App.feModifyElemPropID (0-br, opid)
     'App.feAppMessage(FCM_NORMAL , Str(Ar(br)))
    Wend
     App.feAppMessage(FCM_NORMAL, "Masa nakon optimizacije")
     MacroRun CurDir$() & "\Masa.BAS",Str(elset.ID)
     OK:
    End Sub

    Sub Analiza
     Dim App As femap.model
     Set App = feFemap()
```

```
    Dim Analiza As femap.AnalysisMgr
    Set Analiza = App.feAnalysisMgr
    Dim aID As Long
    Analiza.title = "Staticka analiza"

    Analiza.Solver =FAM_NX_NASTRAN
    Analiza.AnalysisType = FAT_STATIC
    conset = App.Info_ActiveID(FT_BC_DIR)
    If conset = 0 Then
        App.feAppMessageBox(0,"Nema ogranicenja.")
        Exit Sub
    End If
    Analiza.BCSet (0)=conset
    ldset = App.Info_ActiveID (FT_LOAD_DIR)
    If ldset =0 Then
        App.feAppMessageBox(0,    "Nema    aktivnog    seta
opterecenja.")
        Exit Sub
    End If
    Analiza.BCSet (2) = ldset
    Analiza.Output(15) = -1
    Analiza.Output(16) = -1
    aID = Analiza.NextEmptyID
    Analiza.Put(aID)
    Analiza.Analyze(aID)

    While App.Info_Count(FT_OUT_CASE) <1
        Wait(10)
        App.feAppMessage(FCM_NORMAL, "Analiza je zavrsena")
    Wend
End Sub
```

```
Sub DelOut
  Dim App As femap.model
    Set App = feFemap()
    Dim oSet As femap.Set
    Set oSet = App.feSet
    oSet.AddAll( FT_OUT_CASE )
oSet.Reset( )
    While oSet.Next( )
      App.feDelete( FT_OUT_CASE, oSet.CurrentID)
    Wend
    App.feAppMessage( FCM_NORMAL, " ")
End Sub

Sub BLoad

 Dim App As femap.model
    Set App = feFemap()
    ' dodeljivanje sile zemljine teze
 Dim gr As femap.LoadSet
    Set gr = App.feLoadSet
    Dim grID As Long
 grID =Abs( gr.ID)
 gr.BodyAccelOn = True
    gr.BodyAccel(1) = -9810
    gr.Put(grID)
End Sub

Sub ExcelP
 Dim appExcel As Excel.Application
 Set appExcel =  New Excel.Application
```

```
        Dim wrkBook As Excel.Workbook
     Set   wrkBook   =   appExcel.Workbooks.Open(CurDir$()   &
"\femap_pod.xlsx")
     Dim wrkSheet As Excel.Worksheet
     Set wrkSheet = wrkBook.Worksheets(1)
    End Sub

    Sub  PPPR(v As Double, rv As Double, d As Integer)
    ' Dimenzionisanje promenljive za domen
     Dim rng As Range
     d=2
     Set rng = Range("B2: B400")
     For Each cll In rng
          d=d+1
          If cll.value > v Then
               rv=cll.value
               Exit For
          End If
     Next
     Call Analiza
     App.feAppMessage(FCM_NORMAL, "Masa nakon optimizacije")
     MacroRun CurDir$() & "\Masa.BAS",Str(elset.ID)
     OK:
    End Sub
```

ПРИЛОГ 2: API програм за рачунање масе конструкције

```
Sub Main
    Dim App As femap.model
    Set App = feFemap()
 Dim elsetID As Integer
 elsetID = CInt(Command$)
 Dim Len1 As Double
 Dim area As Double
 Dim volume As Double
 Dim structMass As Double
 Dim nonStruct As Double
 Dim TotalMass As Double
    Dim structCG As Variant
    Dim nonstructCG As Variant
    Dim totalCG As Variant
    Dim inertia As Variant
    Dim intertiaCG As  Variant
    App.feMeasureMeshMassProp(elsetID,  0,  False,  False,
Len1, area, volume, structMass, nonStruct, TotalMass, structCG,
nonstructCG, totalCG, inertia, intertiaCG)
    por = Str(Round(TotalMass,6)*1000)+"kg"
   App.feAppMessage (FCM_NORMAL , por)
End Sub
```