Универзитет у Београду

Електротехнички факултет


# Примена генетског алгоритма на проблем трговачког путника

Семинарски рад

Ментор:

др Татјана Лутовац, проф.

Кандидат:

Алекса Цветановић

2018/0160

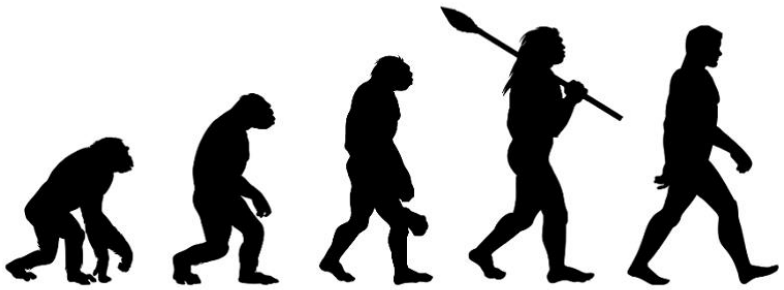
Београд, Децембар 2020.

# Садржај

# Генетски алгоритми

## Увод

Генетски алгоритми представљају изузетно широку и често коришћену класу оптимизационих алгоритама и алгоритама претраге. Свој назив добили су по томе што су инспирисани еволуцијом живог света у природи. Популацију неке врсте чине јединке различитих особина. Због тога су оне различито адаптиране на средину у којој се налазе – неке су боље, а неке лошије прилагођене условима у којима живе. Особине које имају дају јединкама различиту вероватноћу да преживе и оставе потомство. Оне које су боље прилагођене имаће више шанси да оставе потомство, чиме њихов генетски материјал остаје у популацији, док ће оне које се нису прилагодиле условима имати мање шансе да преживе. Такође, у популацији се повремено јављају случајне мутације, које могу повећати или смањити прилагођеност јединке средини. Дакле, еволуција представља одређену врсту оптимизационог процеса, где се генетски материјал на неки начин оптимизује кроз генерације. Генетски алгоритми користе управо овај принцип: на почетку могућа решења проблема моделују се као јединке које чине популацију, на коју се примењују модели процеса као што су укрштања, мутације, селекције и друго.



**Слика 1. Еволуција човека**

Основна карактеристика генетских алгоритама је **општост**. Они спадају у *метахеуристичке методе* – методе за решавање оптимизационих проблема које су настале независно од неког конкретног проблема. Оне описују принцип решавања велике класе проблема, па се исти алгоритам може користити за решавање више различитих проблема. Наравно, некада је неопходно прилагодити алгоритам за конкретан проблем, али је основни принцип исти.

Због своје општости генетске алгоритме одликује и њихова **једноставност**. Разумевање ових алгоритама је скоро интуитивно, јер они потичу из еволуционих принципа познатих свима. Такође, често имплементација ових алгоритама не представља тежак посао, што је веома важно. Међутим, за најбоље резултате често је неопходно прилагодити ове алгоритме конкретном проблему – треба изабрати одговарајући модел за еволуционе процесе, изабрати оптималан број јединки у популацији, добро дефинисати критеријуме за завршетак алгоритма итд.

Главна мана генетских алгоритама јесте **немогућност да увек дају најбоље решење** – током извршавања алгоритма резултат ће се оптимизовати, али због начина на који је услов за завршетак алгоритма дефинисан не може се увек добити глобални оптимум. Услов за завршетак се може дефинисати на разне начине – до настанка одређеног броја генерација, постигнуте довољне прилагођености и сл. Овако дефинисан услов не гарантује да ће излаз алгоритма бити најбоље могуће решење проблема. Штавише, применом генетских алгоритама не може се, у случају проналаска глобалног оптимума, гарантовати да је добијено решење управо оптимум и да не постоји боље решење од добијеног. Упркос овим манама, често за решење није неопходно да буде најбоље могуће, већ да задовољава одређене критеријуме на основу којих се најчешће и дефинише услов за завршетак алгоритма. Због тога генетски алгоритми налазе примену у свим проблемима где није неопходно наћи најбоље могуће решење, већ је довољно задовољити унапред постављене критеријуме.

Иако их карактерише општост, за генетске алгоритме постоји огроман број параметара које унапред треба дефинисати, који зависе од конкретног проблема, и који директно утичу на перформансе алгоритама и квалитет крајњег решења. Да би се добило што боље решење, треба унапред одредити број јединки у популацији, број генерација, начин на који се врши селекција, начин на који се моделује процес мутације и још много тога. Обично је при коришћењу ових алгоритама управо најтеже изабрати погодне параметре за добијање оптималног решења. Најчешће се ови параметри подешавају пропуштањем тест примера кроз алгоритме и упоређивањем добијених резултата. Другим речима, неопходно је алгоритам прилагодити конкретном проблему и улазима у циљу побољшања његових перформанси.

# Компоненте генетског алгоритма

Генетски алгоритми су због своје општости погодни за решавање великог броја проблема, и без великих измена у имплементацији могу се применити на различите конкретне проблеме. Ови алгоритми састоје се из компоненти које су неопходне за њихову реализацију без обзира на проблем који се решава. Све ове компоненте инспирисане су еволуцијом живог света, и због тога је коришћена терминологија најчешће позајмљена из генетике. Такође, при моделовању различитих структура и процеса тежи се ка што већој сличности са еквивалентним структурама и процесима из живог света.

## Репрезентација јединки

*Јединка* у биологији означава један примерак, биће, које је члан веће групе зване *популација*. Свака јединка има свој генетски материјал који се налази у хромозомима њених ћелија. *Хромозоми* се састоје из *гена*, који заједно чине *генотип* јединке. Скуп особина јединке које она има, које су одређене генотипом, назива се *фенотип*. Јединка у генетском алгоритму представља једну инстанцу, могуће решење датог проблема. Информације у њој представљају њен генотип, а особине које има на основу тих информација су фенотип јединке.

Јединка се може представити различитим структурама података, као на пример низом бројева, низом карактера, стаблом, матрицом и другим. Перформансе алгоритма значајно зависе од избора репрезентације јединки, па је за ефикасан алгоритам потребно изабрати што је могуће оптималнију репрезентацију. Такође, неопходно је над изабраном репрезентацијом правилно дефинисати операције као што су укрштање и мутација. При генерисању прве генерације, или при настанку јединке као резултата генетских оператора може се десити да добијена јединка не представља могуће решење проблема. Ови случајеви могу се избећи тако што се операције дефинишу тако да сигурно никад не креирају овакве јединке, или да се, у случају таква јединка креира, она касније поправи тако да може да представља одговарајуће решење.

Најчешће коришћена репрезентација јединке јесте као низ бинарних цифара. Цео низ представља хромозом, односно генотип, а одређена цифра или секвенца цифара њен ген. Вредност садржана у одређеном гену слика се у неку особину јединке, на основу које се процењује прилагођеност, односно квалитет решења.

| 0 | 1 | 0 | 0 | 0 | 1 | 1 | 0 |
|---|---|---|---|---|---|---|---|

**Слика 2. Бинарна репрезентација јединке**

Поред бинарне репрезентације, сличне њој су октална и хексадецимална репрезентација. Аналогно бинарној, ове репрезентације представљају скуп окталних, односно хексадецималних цифара од којих свака цифра представља један ген. Коришћењем ових репрезентација одређене особине се могу много боље квалитативно приказати – ако у бинарној репрезентацији једна цифра представља ген, на овај начин се може осликати само присуство или одсуство неке особине. У октално ј и хексадецималној репрезентацији може се без тешкоћа дефинисати у којој мери је присутна нека особина, колико је она изражена код јединке. Ове две репрезентације на неки начин поседују додатну димензију коју бинарна репрезентација нема.

Приликом решавања проблема трговачког путника јединка је представљена као низ карактера. Оваква репрезентација је погодна за перформансе алгоритма, али и за обраду над генотипом јединке. На овај начин омогућава се ефикасна и једноставна обрада генотипа јединке, с обзиром на његову специфичност у оквиру овог проблема.

## Популација

Скуп јединки чини једну популацију. За њу се може везати велики број параметара. На пример, број јединки у популацији може бити фиксан или променљив. Број јединки у првој генерацији може бити унапред задат, а може бити и случајан у неком опсегу. Популација може да садржи јединке од којих неке имају исти генотип, а може бити дефинисана тако да су све јединке различите. У зависности од конкретног проблема дефинишу се ови параметри, од којих неки мање, а неки више утичу на квалитет добијеног решења.

## Иницијализација

Пре извршавања операција мутације и укрштања, потребно је генерисати прву генерацију јединки. Њу могу чинити јединке чији је генотип генерисан на различите начине – њихови гени могу бити потпуно насумични, а могу бити генерисани и са различитим вероватноћама у зависности од информација о проблему које већ поседујемо. Уколико имамо нека очекивања о решењу, погодним генерисањем прве генерације могу се поправити перформансе алгоритма. Такође, на улаз генетског алгоритма могу се довести и излази неког лошијег алгоритма.

## Политика замене генерације

Иницијална популација представља прву генерацију јединки. Процесима мутације и укрштања настају нове јединке, које су потомци јединки из прве генерације. Као у живом свету, и у генетском алгоритму постоји однос родитеља и потомства који се може имплементирати на различите начине.

Политика замене генерације је веома важна за генетске алгоритме, па је на основу овог параметра извршена њихова подела. У зависности од тога како се имплементира замена генерације генетски алгоритми се могу поделити на генерацијске генетске алгоритме (generational genetic algorithm) и генетске алгоритме стабилног стања (steady state genetic algorithm).

Генерацијске генетске алгоритме одликује замене целе популације одједном по генерацијама. Из текуће генерације процесом селекције бирају се родитељи. Укрштањем родитеља и мутацијом настају јединке нове генерације, које замењују све јединке из претходне.

Генетски алгоритми стабилног стања јединке се у популацији мењају једна по једна. Бирањем родитеља, укрштањем, и мутацијом настаје потомак који мења неку од јединки из популације. Постоје различите политике ове замене:

- *Замена најгорих* – нове јединке замењују најгоре прилагођене јединке у популацији
- *Насумична замена* – нове јединке замењују случајно изабране јединке у популацији
- *Такмичење родитеља и потомака* – уколико су потомци боље прилагођени од родитеља, биће убачени у популацију уместо њих

У случају насумичне замене и сличних политика замене, може се десити да најбоље прилагођене јединке буду замењене, што може утицати на перформансе алгоритма. Због тога се може увести *елитизам* – стратегија која најбољим јединкама у популацији гарантује останак у њој. Такође, елитизам може подразумевати и гаранцију да неће доћи до мутације. Ова стратегија применљива је и у генерацијским алгоритмима и у алгоритмима стабилног стања.

## Функција прилагођености

Једна од најбитнијих компоненти сваког генетског алгоритма јесте функција прилагођености. На основу ове функције одређује се квалитет датог решења. На основу резултата ове функције јединке се рангирају и њихова позиција је битна за скоро све остале компоненте

генетског алгоритма. На пример, на основу ове функције одређују се најгоре јединке које ће бити замењене у случају примене такве политике замене. На основу вредности ове функције за различите јединке може се дефинисати услов за крај алгоритма. Јединке које имају већу прилагођеност могу имати веће шансе за остављање потомства, уколико је на тај начин дефинисан процес селекције. Најважније, у коначној популацији јединка која има најбољу вредност ове функције представља оптимално решење и излаз алгоритма.

Функција прилагођености може имати различите особине – иако су неке пожељне, није неопходно да има ниједну од често тражених особина (нпр. диференцијабилност) за потребе генетског алгоритма. Међутим, неке од особина као што је ненегативност могу знатно олакшати имплементацију алгоритма и побољшати његове перформансе. Веома је битно добро дефинисати ову функцију да би се решења рангирала на прави начин, јер ће то директно утицати на излаз алгоритма.

За проблем трговачког путника је прилично једноставно дефинисати функцију прилагођености. Најбоље решење јесте оно за које путник пређе најмањи пут, тако да сам пређени пут може послужити као вредност функције прилагођености. Наравно, решења ће бити сортирања у растућем поретку по пређеном путу, где ће она са на почетку бити најбоља.

## Селекција

Захваљујући својој прилагођености неке јединке у популацији ће имати веће шансе за репродукију и остављање потомства, док неке, са обично лошијим генетским материјалом, неће имати прилику да тај материјал пренесу на будуће генерације. Овај процес назива се селекција и прилично је еквивалентан у живом свету и генетским алгоритмима.

Иако постоји много принципа селекције, заједничко правило свих њих јесте да јединке са бољим генетским материјалом, и самим тим боље рангирани на основу функције прилагођености, имају веће шансе за остављање потомства. У најједноставнијим моделима најбоље рангиране јединке бивају селектоване и учествују у репродукцији, чиме се веома брзо губе лошији гени популације. Мало комплекснији модели остављају, додуше мању, могућност да лошије рангиране јединке учествују у репродукцији. На тај начин се лошији гени дуже задржавају у популацији, али се може избећи прерано сужавање области у којој се тражи решење, односно може се добити боље решење проблема. Избор начина селекције зависи од типа проблема.

Најпознатији типови селекције јесу *рулетска* и *турнирска селекција.* Рулетску селекцију одликује принцип по коме је вероватноћа јединке да буде изабрана сразмерна њеној вредности функције прилагођености. Све јединке имају шансе да буду изабране, али оне са лошијим генима ће бити изабране са много мањом вероватноћом него оне са бољим. Турнирска селекција је карактеристична по томе што се из популације насумично бира одређен број јединки, од којих она са најбољом вредношћу карактеристичне функције бива изабрана за репродукцију. Такође, уколико је нека јединка раније изабрана за учешће на турниру, може јој се забранити поновно учешће и тако омогућити избор више различитих јединки у турнирима.

## Репродукција

Јединке из популације које су изабране процесом селекције сада учествују у процесу репродукције. Хромозоми две јединке се комбинују, и њиховом комбинацијом настају нове јединке које могу, али и не морају постати чланови популације. Јединке које дају генетски материјал називају се родитељима, а јединке које од тог генетског материјала настају називају се потомцима, односно децом.

Процес репродукције у живом свету је јако компликован, па се у генетским алгоритмима користе поједностављене варијанте. Најчешће се своди на избор једне или више тачака у генотипу једног и другог родитеља, и долази до прекомбинације низа гена на основу тих тачака. Број изабраних тачака може бити различит, али мора бити мањи од величине низа цифара којим се моделује хромозом јединке. Након укрштања родитеља на овај начин настају два потомка, као што је илустровано на слици 3.

| | | | | | |
|---|---|---|---|---|---|
| **Родитељ 1** | **0 1 1** | **0 0 1 0 1** | → | **Потомак 1** | **0 1 1** | **1 0 0 0 0** |
| **Родитељ 2** | **1 0 1** | **1 0 0 0 0** | | **Потомак 2** | **1 0 1** | **0 0 1 0 1** |

**Слика 3. Једнопозиционо укрштање за бинарну репрезентацију**

Укрштањем гена двеју јединки могу настати генотипи који не представљају валидну интерпретацију решења. Овакви случајеви захтевају посебну пажњу и њима се може баратати на различите начине. Најједноставнији начин да се овај случај избегне јесте да се операција укрштања дефинише на тај начин да никад не дође до стварања неисправних јединки. Уколико је то немогуће, при настанку овакве јединке њен генотип се мора поправљати неким механизмима тако да решење буде могуће.

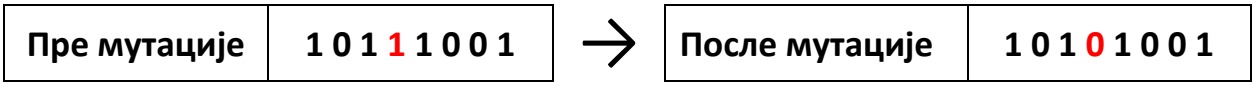
У проблему трговачког путника неопходно је сваки град посетити тачно једном. Због тога постоји велики број непостојећих решења који може настати описаним механизмом укрштања. Овај проблем решава се другачије дефинисаном операцијом укрштања, којом се избегава појава неисправних јединки.

## Мутација

Мутација представља процес у природи где долази до промене гена у оквиру јединке. Мутације су потпуно случајне, и могу утицати добро, лоше, или да немају никакав утицај на прилагођеност јединке средини. Вероватноћа да ће доћи до мутације зависи од типа мутације, али је за сваку јединку једнака.

Процес мутације се може моделовати и у оквиру генетских алгоритама. Наравно, постоје различити начини за то, и избор одговарајућег модела зависи од конкретног проблема који се решава. Мутације уносе различитост у популацију и подстичу обнављање изгубљеног генетског материјала, и могу утицати на то да алгоритам да оптималније решење проблема.

Вероватноћа мутације не сме бити превелика, јер у том случају никада неће доћи до сужавања опсега решења. Међутим, уколико је вероватноћа мутације мала, или до мутације уопште не долази, претрага решења се врло брзо може сузити на неки опсег у коме постоји добро, али не и најбоље решење.




**Слика 4. Пример мутације за бинарну репрезентацију**

### Заустављање

У генетским алгоритмима неопходно је дефинисати услов за њихов завршетак. Овај услов може се дефинисати на различите начине, и свака има своје предности и мане. Избор начина заустављања утиче на перформансе алгоритма, али и на квалитет коначног решења.

Један од услова за заустављање који се користи јесте да најбоља јединка у популацији задовољава неки задати критеријум. Овако се врло често неће добити најбоље решење, али добијено решење ће задовољити тражене особине и алгоритам ће бити од користи. Може се задати фиксан број генерација које треба генерисати пре излаза из алгоритма. На овај начин биће познато трајање рада алгоритма, али квалитет решења неће увек бити задовољавајући. Може се пратити функција прилагођености и промена њене вредности кроз генерације. Услов завршетка алгоритма може бити да неко време не дође до веће промене вредности функције прилагођености за најбоље јединке. Постоји још могућности, али се најчешће користи комбинација неколико услова, јер се тако прави компромис између квалитета решења и перформанси алгоритма.

## Типови генетских алгоритама

За разлику од многих алгоритама претраживања који за исправан рад траже унапред задату што већу количину информација, за генетске алгоритме би се могло рећи да раде „на слепо“ – за њихову имплементацију нису потребне никакве додатне информације сем вредности функције циља придружене јединкама које се испитују. Ова карактеристика генетске алгоритме чини прикладнијим од великог броја других алгоритама претраживања, за које у старту не постоји довољно информација да би дали користан резултат.

## Општи генетски алгоритам (ОГА)

Постоји више типова генетских алгоритама. Међутим, сви они се ослањају на *општи генетски алгоритам*. Овај алгоритам представља основу за све остале генетске алгоритме, који се добијају његовом модификацијом. Овај алгоритам је веома једноставан, и дат је тако да се може применити на велики број различитих проблема.

Општи генетски алгоритам је следећи:

1. Генериши почетну популацију јединки
2. Израчунај прилагођеност сваке јединке у популацији
3. Понављај
    a. Изабери из популације скуп јединки за репродукцију
    b. Применом оператор укрштања и мутације креирај нове јединке
    c. На основу старих и нових јединки креирај нову генерацију
4. док није испуњен услов заустављања
5. Врати најквалитетнију јединку у последњој популацији

За решавање конкретног проблема потребно је дефинисати функције које су коришћене у алгоритму, и то најчешће представља најтежи део у решавању проблема, док је сам алгоритам и имплементација једноставна. Општи генетски алгоритам је погодан у следећим ситуацијама:

- Почетне информације о проблему су тешко разумљиве
- Почетно знање о проблему је веома мало
- Немогућа је било каква математичка анализа улазних података

## Паралелни генетски алгоритам (ПГА)

Једна од великих мана општег генетског алгоритма јесте његово време извршавања. Ово време се може скратити применом паралелног генетског алгоритма, који заправо представља више општих генетских алгоритама који се извршавају истовремено. Овакав алгоритам није тешко имплементирати, и има значајно боље перформансе. Постоји више различитих врста ПГА. Разлике између њих нису велике, и углавном се односе на комуникацију између субпопулација различитих ОГА који се извршавају.

Најпростији начин имплементације ПГА јесте *независни ПГА,* и он се своди на извршавање више идентичних ОГА истовремено. Сваки од ОГА почиње са различитим делом популације, и извршава се потпуно независно од осталих. Није дозвољена комуникација између субпопулација које су улази у различите ОГА. Алгоритам ПГА је завршен када сваки од ОГА да резултат. На овај начин се у потпуности смањује могућност да услед мешања јединки различитих субпопулација у више ОГА дође до конвергенције података ка неком решењу, које није потпуно оптимално.

Још једна врста ПГА јесте *ПГА са миграцијом*, где је дозвољено да понекад нека јединка пређе из једне субпопулације у другу. Овај процес назива се *миграција* и њиме се моделује миграција која је присутна и у живом свету. Миграција се одвија након унапред задатог броја итерација, тако што се хромозом најбоље јединке из сваке субпопулације копира и шаље у једну од

осталих субпопулација. Најбоља јединка из прве субпопулације мигрира у другу, друга у трећу, и тако редом. Оне замењују најгору јединку из сваке субпопулације.

## Адаптивни генетски алгоритам (АГА)

Општи генетски алгоритми подразумевају унапред дефинисане операције укрштања, мутације и осталих. Адаптивне генетске алгоритме карактерише промена параметара везаних за ове операције у току самог извршавања алгоритма. На пример, током извршавања АГА може се мењати вероватноћа мутације сваке јединке: што популација дуже не напредује, то је већа вероватноћа мутације, и обратно.

Вероватноћа укрштања и вероватноћа мутације су најбитнији за балансирану претрагу у читавом домену решења. Уколико су они лоше изабрани може доћи ка прераној конвергенцији популације ка неком решењу које није оптимално. Због тога се уводи промена ових параметара уколико тренутни резултати нису задовољавајући, да би се у популацију увео нов генетски материјал и претрага проширила на што већи скуп решења.

Поред ових типова генетских алгоритама постоје још многи. Иако су сви слични и потичу од ОГА, њихове разлике, иако не велике, могу изузетно утицати на квалитет решења. Избор одговарајућег генетског алгоритма зависи од конкретног проблема, и пре решавања проблема неопходно је спровести тестове са различитим типовима генетског алгоритма.

# Проблем трговачког путника

## Историја и формулација

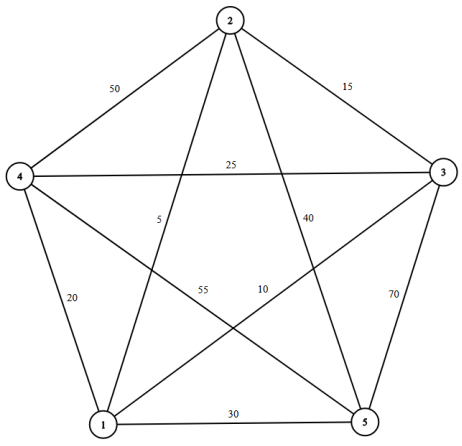
Проблем трговачког путника је један од најпознатијих математичких проблема. Први пут је математички формулисан у деветнаестом веку од стране ирског математичара Хамилтона и британског математичара Киркмана. Овај проблем постаје веома популаран од педесетих и шездесетих година двадесетог века, и до данас се проучава у оквиру различитих области математике и рачунарства.

Проблем трговачког путника формулисан је на следећи начин:

*Ако је познат скуп градова и удаљеност између свака два града, који је најкраћи могући пут којим се сваки град обилази тачно једном и враћа у почетни град?*

Проблем трговачког путника јесте **минимизациони проблем**, где дужина пута који трговачки путник треба да пређе треба да буде минимална. Овај је квадратне сложености, што значи да ће у најгорем могућем случају за N градова бити потребно $N^2$ операција. Треба приметити да за случај N градова може постојати више решења са минималном дужином пута, која се разликују само по томе који је почетни град. Градови се посећују истим редоследом, само се полази из различитих градова.

Овај проблем је најбоље моделовати као неоријентисан граф, где чворови представљају градове, а гране путеве. Свака грана има своју тежину, која представља удаљеност од једног града до другог. Граф може бити потпун, чиме се подразумева да између свака два града постоји пут.




**Слика 5. Приказ проблема трговачког путника у виду графа**

# Различите варијанте проблема трговачког путника

Из проблема трговачког путника настало је више различитих проблема који представљају његову генерализацију или конкретизацију. Неки од критеријума по којима се овај проблем може разликовати су:

- Симетричност
- Потпуност

У погледу симетричности овај проблем се може посматрати као симетричан, где је дужина пута између оба града иста у оба смера, и асиметричан, где је дужина пута различита у зависности од тога у ком смеру се путник креће. Потпуност подразумева то да ли постоји пут између свака два града. Уколико не подразумевамо потпуност, немогуће је из сваког града стићи у било који други, те се то мора узети у обзир. У том случају до неког града се може стићи само из одређених градова, што може искључити решења која би потенцијално била оптималнија да пут постоји.

Једна од општијих верзија овог проблема јесте *проблем путујућег купца*, који се бави купцем који купује производе у различитим градовима. Производе може купити у различитим градовима, али по различитим ценама, и нема сваког производа у свим градовима. Циљ је наћи пут између подскупа градова тако да купац купи све производе и минимизује потрошњу, коју чине цене производа и путни трошкови.

# Методе решавања проблема

Проблем трговачког путника је познат у математици и рачунарству, и сходно томе постоје различити начини његовог решавања. Такође, његово решавање зависи и од његових параметара, као што је број градова.

Као најприроднije решење намеће се *насилна претрага*, где се генерише свако могуће решење и врше поређења док се не дође до оптималног решења. Потребно је генерисати све могуће пермутације градова и извршити велики број поређења, што овај алгоритам чини веома захтевним и лоших перформанси. С обзиром да је његова сложеност O(N!) овај метод постаје непрактичан за веома мали број градова.

Постоји више модификованих верзија овог алгоритма, чиме се побољшавају перформансе алгоритма и повећава број градова за које се решење може наћи за задовољавајуће време. Међутим, упркос развоју рачунарских процесора и повећању брзина на којима раде, и даље су за већи број градова ови алгоритми веома спори и непрактични.

Да би се смањило време израчунавања неопходно је користити алгоритме који уводе неке апроксимације. На овај начин се време решавања значајно смањује, али се не може гарантовати најбоље могуће решење. Један од најпознатијих алгоритама за апроксимативно решавање јесте *NN алгоритам*, који се заснива на томе да трговац за следећи град увек бира онај који је најближи

граду у коме се он тренутно налази. На овај начин до решења се долази знатно брже, и у просеку оно даје пут око 25% дужи од најкраћег могућег пута.

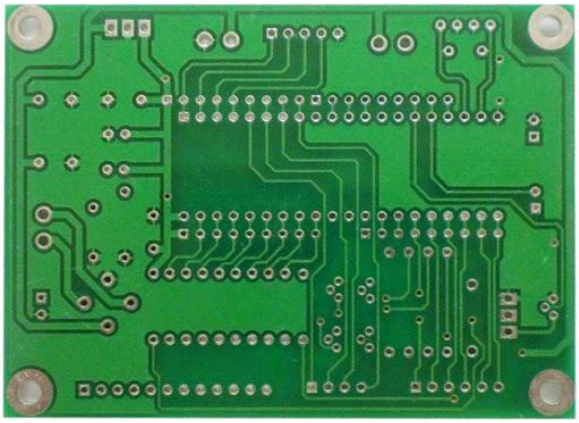
Један од најзанимљивијих алгоритама који се заснива на апрокисмацијама инспирисан је колонијом мрава, па по њој носи име *Ant Colony Algorithm*. Алгоритам има за узор праве мраве који у потрази за најближим путем између гнезда и хране испуштају феромоне на основу којих ће се касније управљати. Модели мрава у алгоритму памте путеве по којима су се раније кретали, да би на основу њих пробали да нађу краћи пут.

Проблем трговачког путника може се решавати неким од алгоритама из класе генетских алгоритама. Решења добијена применом овог алгоритма нису нужно најбоља могућа, али могу бити веома добра. Такође, ове алгоритме је једноставно имплементирати у односу на неке друге класе алгоритама, па је њихова примена у решавању овог проблема сасвим оправдана.

## Реални проблеми

Мотивација за решавање проблема трговачког путника јесте управо његова примена у реалном свету. Трговци, али и путујући циркуси, музичари, поштари, возачи аутобуса, само су неки од људи којима овај проблем представља, или је у историји представљао саставни део живота. Кад год је у питању неко путовање или обилазак, намеће се питање коју руту изабрати да би путни трошкови били минимални. На пример, при одржавању школских екскурзија пожељно је провести што мање времена у путу и обићи што више градова.

Резултати овог проблема не морају бити стриктно везани за путовања и градове. На пример, у процесу израде штампаних плоча пожељно је да бушилица којом се буше рупе на колима прелази најкраћи пут до сваке рупе. Такође, приликом коришћења 3D штампача, да би време штампања било што мање, глава штампача треба да пређе најкраћи пут приликом штампања. Евидентно је да је решење проблема трговачког путника корисно у различитим областима.



**Слика 6. Рупе на штампаној плочи**

# Решавање проблема трговачког путника применом генетског алгоритма

Због природе проблема трговачког путника проналажење оптималног решења неким алгоритмом који даје најкраћи пут би за велики број градова трајало веома дуго. Због тога је за већи број градова сасвим оправдано користити неку од метода која не тестира сваки могући распоред градова, чиме би се време извршавања алгоритма знатно смањило. Решење које генетски алгоритам даје готово сигурно неће бити најбоље могуће, али уз добро дефинисан услов за завршетак алгоритма може се доћи до задовољавајућег решења.

У имплементацији решења коришћен је општи генетски алгоритам због своје једноставности. Иако би се применом неког другог генетског алгоритма могло добити боље решење, на примеру ОГА сасвим је могуће приказати компоненте које већина генетских алгоритама има, њихов значај и утицај на коначно решење.

Имплементација је написана на програмском језику C++, због разумљивости овог језика, и због тога што је веома једноставно имплементирати све потребне компоненте генетског алгоритма. Такође, овај програмски језик је знатно бржи у случају одређивања решења за већи број градова, већи број јединке у популацији, или већи број генерација које треба створити. Комуникација са корисником омогућена је преко стандардног улаза и излаза, али је у код уграђена могућност коришћена подразумеваних вредности, које су коришћене и за демонстрацију рада написаног програма.

Применом алгоритма тражи се најкраћи пут за обилазак 7 европских престоница – Београда, Берлина, Лондона, Мадрида, Москве, Париза и Рима. Подразумева се полазак из Београда. Такође, сматра се да је пут од једног до другог града исте дужине у оба смера, и да између свака два града постоји пут, иако је у код уграђена и могућност да пут између нека два града не постоји. Посматра се 15 генерација, од којих свака има по 20 јединки. Кроз сваку генерацију памти се најбоља јединка и она се упоређује са најбољом јединком следеће генерације, и за коначно решење узима се најбоља јединка из свих 15 генерација.

# Дефинисање компоненти генетског алгоритма

## Репрезентација јединки

Свака јединка представља редослед градова који се обилазе. Проблем трговачког путника захтева да овај редослед има одређене особине, иначе се решење сматра некоректним. Свака јединка треба да почиње и завршава се истим градом. У имплементацији за почетни град је изабран Београд, иако у самом проблему није тачно прецизирано од ког града би морало да се крене. Такође, мора се обићи сваки град тачно једном, што значи да јединка мора да садржи све градове и то тачно једном.

Свака јединка моделована је као структура која се састоји из низа карактера и целобројне променљиве. Низ карактера (стринг) садржи бројеве који означавају градове, на следећи начин:

- 0 – Београд
- 1 – Берлин
- 2 – Лондон
- 3 – Мадрид
- 4 – Москва
- 5 – Париз
- 6 – Лондон

Сваки стринг садржи тачно 8 карактера, од којих су први и последњи 0, што представља полазак и повратак у Београд, како је дато у формулацији проблема. Целобројна променљива садржи информацију о прилагођености јединке, која се користи при процесу укрштања, мутације и самом избору коначног решења.

| 0 | 2 | 4 | 6 | 5 | 3 | 1 | 0 |
|---|---|---|---|---|---|---|---|

**Слика 7. Пример генотипа јединке у имплементацији проблема трговачког путника**

## Популација

Популација представља скуп јединки организоване као вектор. У вектору се јединке сортирају по прилагођености, на основу чега касније долази до мутације и селекције. Популација је моделована тако да у сваком тренутку садржи исти број јединки, и да у њој могу да буду јединке из различитих генерација

## Иницијализација

Иницијална популација креира се потпуно насумично. Наравно, постоје уграђени механизми за отклањање непостојећих решења, као што су она са дуплираним бројевима. Свака јединка почиње бројем 0, након чега се насумично генеришу бројеви од 1 до 6 и, уколико се не налазе у тренутном стању генотипа јединке, додају се у низ карактера. Након 6 бројева тако додатих у низ карактера на самом крају се поново додаје број 0, што означава повратак у Београд.

## Политика замене генерације

Генетски алгоритам који је примењен при решавању проблема трговачког путника је генетски алгоритам стабилног стања, што значи да популација у неком тренутку може да се састоји из јединки које потичу из различитих генерација. Такође, операције укрштања и мутације су имплементиране тако да након настанка потомака долази до такмичења између родитеља и потомака. Уколико потомак има лошију вредност функције прилагођености од родитеља, неће бити изабран за нову популацију. Међутим, да би се задржао диверзитет у генетском материјалу, мутација је имплементирана тако да упркос лошијем резултату потомка постоји вероватноћа да он замени родитеља.

Такође, у овом коду постоји и имплементација елитизма – најбоља јединка из сваке генерације са сигурношћу ће бити присутна у следећој генерацији. Ова особина гарантује да ће у последњој генерацији бити присутна најбоља до сада креирана јединка.

## Функција прилагођености

Циљ алгоритма јесте да врати пут најмање дужине којим се обилазе сви градови. Природно се намеће да функција прилагођености враћа дужину пута карактеристичну за сваку јединку, што је у овој имплементацији и примењено. На овај начин се функција прилагођености веома једноставно дефинише, и у потпуности се поклапа са захтевима проблема.

Треба обратити пажњу на рангирање јединки на основу ове вредности. Циљ је да дужина пута буде најмања, што значи да су јединке са мањом вредношћу прилагођености боље од оних чија је вредност прилагођености већа. Није потребно мењати начин одређивања прилагођености да би се добила адекватна веза овог параметра и квалитета генотипа јединке, већ се довољно само тражити што мању вредност овог параметра и јединке распоредити у растућем поретку прилагођености.

## Селекција и репродукција

Проблем трговачког путника је веома специфичан у погледу решавања генетског алгоритма због услова да се сваки град мора обићи тачно једном. Овај услов значајно отежава моделовање репродукције. Из тог разлога, у овој имплементацији репродукција није моделована. Постоје модели којима захтев може бити задовољен, као на пример *cycle crossover,* али су они тешки за имплементацију уколико је генотип представљен низом карактера, као што је овде случај. Међутим, да би решавање проблема трговачког путника на овај начин имало смисла, процес мутације је изузетно добро дефинисан и овај процес се одвија чешће него што би се одвијао у другим имплементацијама. Селекција је моделована у оквиру процеса мутације, јер је неопходна за конвергенцију резултата ка неком крајњем решењу.

## Мутација

За дефинисање процеса мутације потребно је пазити на исти захтев као код репродукције. Такође, услед одсуства репродукције неопходно је да се мутација дешава чешће, како би се створиле нове јединке. У овој имплементацији представљено је веома елегантно решење којим се не допушта настанак јединки које не би могле да представе коректно решење.

Пре процеса мутације свака јединка представља могуће решење, односно њен генотип садржи сваки град тачно једном. Понављање градова се може избећи простом заменом градова на различитим позицијама. Насумично се бирају два града која нису на првом или последњем месту (нису 0) и они мењају своја места.

Након мутације проверава се да ли је новонастала јединка боље прилагођена од њеног родитеља. Уколико јесте, одмах се додаје у вектор који представља нову популацију. Уколико није, постоји одређена вероватноћа да ће она бити додата упркос лошијој прилагођености, да би се задржала различитост генетског материјала и спречила прерана конвергенција алгоритма ка неком решењу. У имплементацији је усвојено да постоји 10% шансе да лошија јединка постане члан популације, и ова вредност је константна током извршавања алгоритма. За овај тест пример овај поступак не доводи до значајнијих промена, али за већи број градова може битно утицати на квалитет решења.

Процес мутације одвија се док свака јединка мутира тако да њена прилагођеност буде боља, или док без мутације буде уврштена у нову популацију. На овај начин компензује недостатак репродукције, и ствара се нови генетски материјал. Такође, обезбеђује се да нова популација има исти број јединки као и претходна.

| **Пре мутације** | **0 2 4 6 5 3 1 0** | → | **После мутације** | **0 3 4 6 5 2 1 0** |
|---|---|---|---|---|

**Слика 8. Пример мутације у имплементацији проблема трговачког путника**

## Заустављање

Као услов заустављања изабрано је да се алгоритам заустави након тачно 15 генерација, због једноставности имплементације овог услова. Такође, веома је лако имплементирати случај да се тражи решење чија је дужина пута изнад неке фиксне вредности, али због прикладније демонстрације резултата овакав услов није дефинисан.

# Резултати

Након пуштања тест примера кроз написани програм добијени су следећи резултати:

```
Inicijalna populacija:
GENOTIP   PRILAGODJENOST
06513240   10045
02634150   11850
05436120   12531
02164350   12112
06354210   10049
01534620   11862
06531420   10544
01436520   10542
04263150   11192
03541620   11475
04325160   9533
02136450   12153
06342510   10236
06234510   11216
01362540   10196
05431620   13539
01634250   11265
06342150   11272
02354610   11046
06345120   11502
```

```
Generacija 1
GENOTIP   PRILAGODJENOST
04325160   9533
06153240   9306
06351240   9152
01462530   9833
06432510   10014
04136520   9683
06351420   9808
05324610   10816
04253160   9374
01234560   10944
03614250   10462
06324150   9775
02541630   10692
04365120   11112
02364150   10618
02534610   11076
02154360   11502
02536410   9425
```

```
05416320   11034
03451620   13129

Generacija 2
GENOTIP   PRILAGODJENOST
06351240   9152
04153260   8665
04352160   9376
02516430   11927
04352160   9376
04156320   9615
06321450   9815
06351240   9152
01465230   9718
06452310   10052
03514260   10216
02365140   9615
06541230   10141
04325610   10040
05234160   9999
01456320   10512
04532610   10125
```

| | |
|---|---|
| 04635120 | 9995 |
| 04152360 | 7882 |
| 03251640 | 9774 |

Generacija 3

| GENOTIP | PRILAGODJENOST |
|---|---|
| 04152360 | 7882 |
| 04163250 | 8911 |
| 06354210 | 10049 |
| 06531240 | 9888 |
| 04253160 | 9374 |
| 04356120 | 11109 |
| 04126350 | 9540 |
| 06325140 | 7882 |
| 01645230 | 10671 |
| 03256140 | 9237 |
| 06324150 | 9775 |
| 04632150 | 9962 |
| 05236140 | 8911 |
| 04325160 | 9533 |
| 06425310 | 9855 |
| 04632510 | 8926 |
| 05641230 | 10754 |
| 03214560 | 10141 |
| 01654320 | 12163 |
| 04516230 | 10976 |

Generacija 4

| GENOTIP | PRILAGODJENOST |
|---|---|
| 04152360 | 7882 |
| 06345120 | 11502 |
| 04153260 | 8665 |
| 05234160 | 9999 |
| 04652310 | 9662 |
| 03652140 | 9083 |
| 04235160 | 9306 |
| 04352160 | 9376 |
| 04123650 | 9425 |
| 05324160 | 9772 |
| 06415320 | 9586 |
| 06532140 | 8393 |
| 04632510 | 8926 |
| 06453210 | 9827 |
| 05214360 | 9845 |
| 01654230 | 11561 |
| 03641250 | 10086 |
| 04615230 | 9774 |
| 04653120 | 10731 |
| 01652340 | 10040 |

Generacija 5

| GENOTIP | PRILAGODJENOST |
|---|---|
| 04152360 | 7882 |
| 06235140 | 8665 |
| 04123560 | 8393 |
| 04362510 | 10158 |
| 06352140 | 7725 |
| 04532160 | 9346 |
| 04152360 | 7882 |
| 04163250 | 8911 |
| 06415230 | 9211 |
| 04152360 | 7882 |
| 05321460 | 9396 |
| 02615430 | 13129 |
| 06451230 | 10678 |
| 05614320 | 11732 |
| 04235160 | 9306 |
| 04652310 | 9662 |
| 03241650 | 11130 |
| 04623150 | 10970 |
| 06315420 | 11500 |
| 01634250 | 11265 |

Generacija 6

| GENOTIP | PRILAGODJENOST |
|---|---|
| 06352140 | 7725 |
| 04152630 | 9355 |
| 04125360 | 7725 |
| 04125360 | 7725 |
| 04125360 | 7725 |
| 06532140 | 8393 |
| 01463250 | 9392 |
| 06435210 | 9857 |
| 04215360 | 9152 |
| 04132560 | 8618 |
| 03521460 | 9054 |
| 04152360 | 7882 |
| 04162530 | 9352 |
| 06251430 | 10443 |
| 04621350 | 10813 |
| 03641250 | 10086 |
| 04631250 | 10030 |
| 06415320 | 9586 |

05641320 11354

02315460 11278

Generacija 7

GENOTIP PRILAGODJENOST

06352140 7725

04521360 9417

04165320 9612

04521360 9417

01452360 8779

06352140 7725

06132540 9571

01523460 10014

04125360 7725

04132560 8618

04132650 10433

02463150 12113

06415230 9211

06415230 9211

04136250 9765

06341250 9845

06251340 10387

04521360 9417

02415360 9808

04651320 10888

Generacija 8

GENOTIP PRILAGODJENOST

06352140 7725

06152340 9533

04126350 9540

04135260 8733

01425360 8582

06412530 9054

06412530 9054

04251360 9377

04321560 10047

04125360 7725

01632540 9342

04125360 7725

04163250 8911

03415260 10443

06314250 9843

06523410 9477

06152340 9533

04123650 9425

04651230 10288

02463510 10852

Generacija 9

GENOTIP PRILAGODJENOST

06352140 7725

04625310 9777

04165320 9612

04125360 7725

04132560 8618

04153260 8665

06312540 9417

06512430 11598

01432560 9477

04251630 9996

04125630 9083

06532410 9250

06352140 7725

04152360 7882

04123650 9425

06413250 9621

04231560 10045

04653210 9437

03415620 12176

06423510 9787

Generacija 10

GENOTIP PRILAGODJENOST

06352140 7725

04215360 9152

01352640 9777

04125360 7725

04532160 9346

04123560 8393

04125360 7725

06432510 10014

06512340 10047

04125630 9083

04653120 10731

01423560 9250

04125360 7725

06412350 9396

04623510 9709

06523410 9477

04253610 9145

04132560 8618

06152430 11084

| | |
|---|---|
| 01435620 | 11325 |

Generacija 11

| GENOTIP | PRILAGODJENOST |
|---|---|
| 06352140 | 7725 |
| 05124360 | 11272 |
| 04165320 | 9612 |
| 05124360 | 11272 |
| 04125360 | 7725 |
| 04123560 | 8393 |
| 04125360 | 7725 |
| 01253640 | 8769 |
| 04125360 | 7725 |
| 04123560 | 8393 |
| 04235160 | 9306 |
| 06214350 | 10628 |
| 04523610 | 9342 |
| 04523610 | 9342 |
| 01253640 | 8769 |
| 01432560 | 9477 |
| 06532140 | 8393 |
| 04153620 | 9730 |
| 01652430 | 11591 |
| 01436520 | 10542 |

Generacija 12

| GENOTIP | PRILAGODJENOST |
|---|---|
| 06352140 | 7725 |
| 05124360 | 11272 |
| 04152360 | 7882 |
| 06125340 | 9376 |
| 04123650 | 9425 |
| 04123650 | 9425 |
| 06352140 | 7725 |
| 06253140 | 8733 |
| 06253140 | 8733 |
| 04135260 | 8733 |
| 01523640 | 8926 |
| 03524610 | 10474 |
| 01423560 | 9250 |
| 04163520 | 8944 |
| 04253610 | 9145 |
| 01432560 | 9477 |
| 05214360 | 9845 |
| 05214360 | 9845 |
| 06124350 | 11269 |
| 01654230 | 11561 |

Generacija 13

| GENOTIP | PRILAGODJENOST |
|---|---|
| 06352140 | 7725 |
| 03652140 | 9083 |
| 04156320 | 9615 |
| 06523140 | 8618 |
| 06352140 | 7725 |
| 04132560 | 8618 |
| 01563240 | 9816 |
| 04163250 | 8911 |
| 06253410 | 9592 |
| 01623540 | 10125 |
| 06123540 | 9346 |
| 04523610 | 9342 |
| 04126350 | 9540 |
| 01432650 | 11292 |
| 05216340 | 10408 |
| 03214560 | 10141 |
| 03521640 | 9617 |
| 06142350 | 9772 |
| 05123460 | 11050 |
| 01254630 | 10517 |

Generacija 14

| GENOTIP | PRILAGODJENOST |
|---|---|
| 06352140 | 7725 |
| 03652140 | 9083 |
| 06325140 | 7882 |
| 04123560 | 8393 |
| 04153260 | 8665 |
| 06352140 | 7725 |
| 01523640 | 8926 |
| 05123640 | 9962 |
| 04123650 | 9425 |
| 06453210 | 9827 |
| 02156340 | 11112 |
| 01523640 | 8926 |
| 06142530 | 9430 |
| 02563140 | 9683 |
| 01653240 | 9813 |
| 03241560 | 10101 |
| 04216350 | 10181 |
| 05214630 | 10086 |
| 03125460 | 10746 |
| 06432150 | 11050 |

| | | |
|---|---|---|
| | 03625140 9355 | 01325460 10052 |
| Generacija 15 | 04163250 8911 | 06234150 10785 |
| GENOTIP PRILAGODJENOST | 06412530 9054 | 02153640 9995 |
| 06352140 7725 | 02163540 10411 | |
| 06452130 10746 | 01253640 8769 | ---------- |
| 04325160 9533 | 04653210 9437 | |
| 04321560 10047 | 06123540 9346 | |
| 04123560 8393 | 01254630 10517 | NAJBOLJA JEDINKA |
| 01253640 8769 | 05241360 9843 | GENOTIP PRILAGODJENOST |
| 01253640 8769 | 04236150 10338 | 06352140 7725 |

Посматрањем вредности функције прилагођености кроз генерације може се приметити просечно побољшање, што је јасан показатељ да се применом генетског алгоритма могу добити жељени резултати. Најбољи резултат јесте секвенца бројева **0-6-3-5-2-1-4-0**, која одговара секвенци градова ***Београд – Рим – Мадрид – Париз – Лондон – Берлин – Москва – Београд***. Ова јединка први пут је настала у 11. генерацији, и захваљујући елитизму остала је до краја алгоритма. Такође, елитизам је заслужан за много боље просечне резултате, што се може показати покретањем кода без ове особине.

Најлошији резултат добијен је у иницијалној популацији, где је добијена секвенца 0-5-4-3-1-6-2-0. Иако су мутације насумичне, због имплементације процеса селекције кроз генерације се у популацији задржавају бољи гени, док они лошији временом ишчезавају. Као резултат се види смањење просечног дужина пута кроз генерације. Могло се десити да се захваљујући мутацији најлошији резултат добије у некој од наредних генерација, али би врло вероватно тај резултат већ кроз следећу генерацију нестао.

# Закључак

Генетски алгоритми представљају моћан алат за решавање великог броја проблема. Карактеристике ових алгоритама омогућавају њихову применљивост на различите типове проблема. Њиховом применом може се знатно брже доћи до повољних резултата у односу на неке друге алгоритме. Са друге стране, њихово решење често није најбоље могуће, али уколико је сам алгоритам добро имплементиран резултати су често задовољавајући.

Генетски алгоритми се могу, као што је показано, применити на проблем трговачког путника уз сасвим задовољавајуће резултате. Њиховом применом се у случају великог броја градова време извршавања алгоритма може знатно смањити, уз адекватан крајњи резултат. Иако перформансе алгоритма зависе од начина имплементације, може се рећи да су генетски алгоритми знатно ефикаснији од неких других алгоритама за решавање овог проблема, као што је насилна претрага.

Иако су доста једноставни за разумевање, генетски алгоритми се могу међусобно разликовати по детаљима који знатно утичу на резултат. Због тога је неопходно пре прихватања решења прилагодити алгоритам проблему, односно изабрати прави генетски алгоритам и начине моделовања процеса и операција које он користи. Ово је често најтежи део имплементације генетског алгоритма, и треба му посветити највише пажње да би се на крају добио најбољи резултат.

# Литература:


1. П. Јаничић, М. Николић: *Вештачка интелигенција*, Математички факултет, 2020.
2. S.N.Sivanandam, S.N.Deepa: *Introduction to Genetic Algorithms,* Springer, 2008.
3. David L. Applegate, Robert E. Bixby, Vasek Chvatal, William J. Cook, *The Traveling Salesman Problem – A Computational Study,* Princeton University Press, 2006.
4. *Genetic Algorithm for Traveling Salesman Problem with Modified Cycle Crossover Operator*, [Online]. Available: https://www.hindawi.com/journals/cin/2017/7430125/ (5.12.2020.)
5. *Traveling Salesman Problem using Genetic Algorithm*, [Online]. Available: https://www.geeksforgeeks.org/traveling-salesman-problem-using-genetic-algorithm/ (5.12.2020.)
6. *Estimated Travel Distance between European Cities* [Online]. Available: http://www.mapcrow.info/european_travel_distance.html (5.12.2020.)