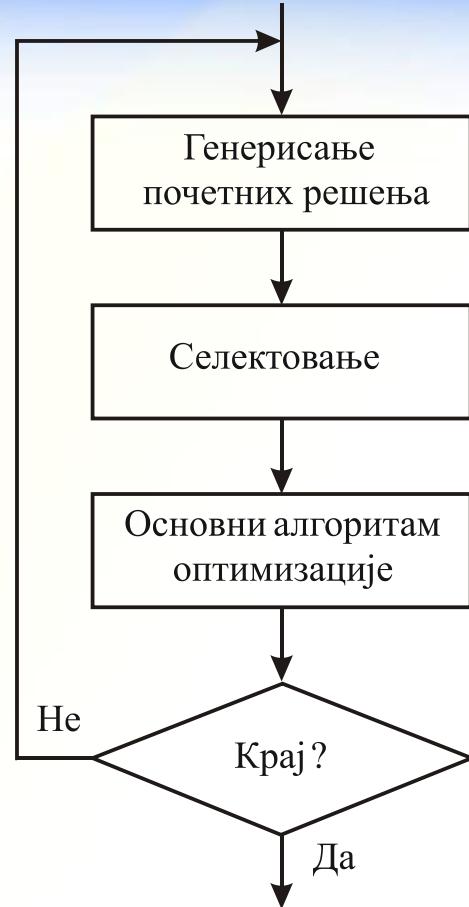


Методе више минимума засноване на понављању

- У пракси, оптимизациони алгоритми се покрећу више пута са различитим почетним решењима
- Такав приступ се примењује уколико је:
 - потребно пронаћи више различитих решења или
 - доказати да больих решења (највероватније) нема
- Три основна корака

Блок дијаграм

- Генерисање скупа почетних решења
- Селекција (нај)бољих решења
- Оптимизација са полазним решењем добијеним у претходном кораку



Практични примери: случајно полазно решење

- Понављање основних оптимизационих алгоритама са случајно изабраном почетном тачком
 - градијентна метода или симплекс са случајно изабраним полазним решењем
 - симулирано каљење са случајном полазном тачком
 - генетички алгоритам са случајно изабраном првом генерацијом
 - PSO или диференцијална еволуција са случајно изабраним полазним популацијама

Практични примери: скуп случајно одабраних решења

- Понављање основних оптимизационих метода са најбољом тачком из претходно генерисаног скупа
 - комбиноване двостепене методе:
 - локални оптимизациони алгоритам покренут са најбољим решењем из случајно генерисаног скупа
 - децимација прве генерације ГА
(прва генерација има знатно више решења од N_{pop} , изаберу се најбоља решења)

Вишеструка понављања: литература и примена

- Вишеструко понављање ГА се у литератури често назива микро-генетички алгоритам (microgenetic algorithm)
- Вишеструко понављање симулираног калења се назива прекаливање (reannealing)
- Ови алгоритми се могу користити за:
 - проналажење једног (глобалног) оптимума и
 - проналажење више решења

Особине поновљених оптимизација

- Добра страна метода заснованих на понављању је лака имплементација
- **Основна мана алгоритама ове класе је недостатак корелације између појединачних покретања**
- Ово има за последицу то да су **крања решења** која се добијају после појединачних покретања алгоритама **међусобно независна**
- Вероватноћа **проналажења истог решења** је релативно велика, што је неефикасан начин оптимизације

Теоријска ефикасност алгоритама заснованих на понављању

- Претпоставимо да оптимизациона функција има N минимума
 - унiformно распоређених у простору и
 - једнаких дубина
- Вероватноћа проналажења сваког минимума је стога једнака и износи $1/N$
- Колика је вероватноћа проналажења свих N минимума после k независних покушаја?

Вероватноћа проналажења свих N минимума из k покушаја, $P(N,k)$

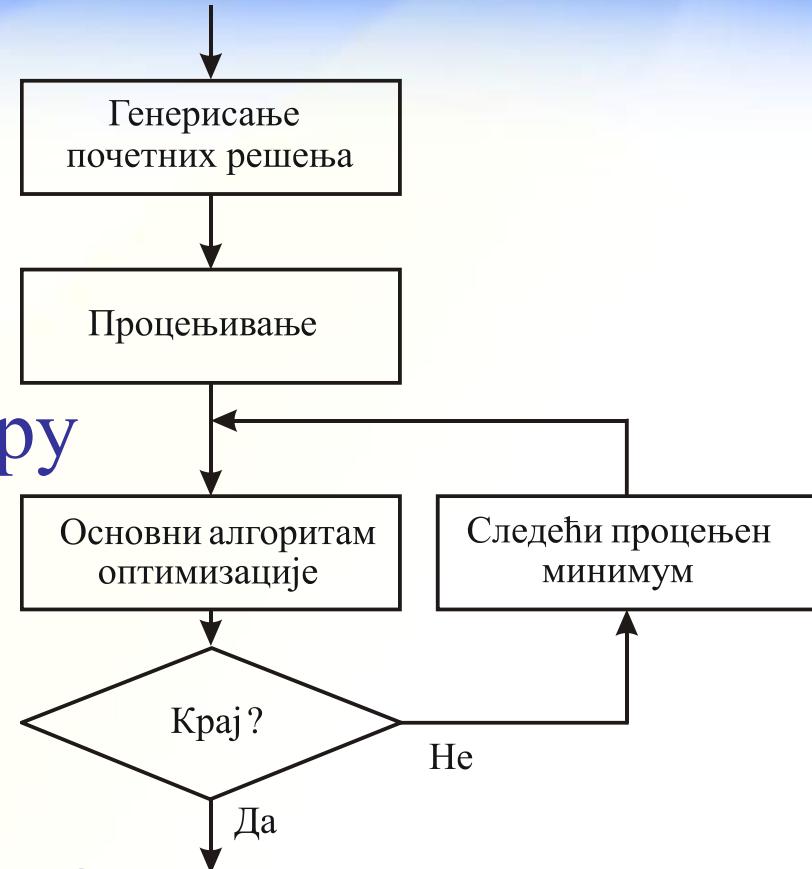
$$P(N,k) = \frac{Q(N,k)}{N^k}$$

$$Q(N,k) = \begin{cases} N^k - \sum_{p=1}^{N-1} \binom{N}{p} \cdot Q(p,k), & N > 1 \\ 1, & N = 1 \end{cases}$$

$N = 10$	$N = 20$	$N = 30$
$P(10, 10) = 3.63 \cdot 10^{-4}$	$P(20, 20) = 2.32 \cdot 10^{-8}$	$P(30, 30) = 1.33 \cdot 10^{-12}$
$P(10, 27) = 0.520$	$P(20, 67) = 0.500$	$P(30, 113) = 0.503$
$P(10, 44) = 0.905$	$P(20, 103) = 0.902$	$P(30, 168) = 0.903$

Процена локалних минимума

- Генерирање полазног скупа решења која се налазе у целом оптимизационом простору
- Процењивање положаја локалних минимума
- Покретање “основног” оптимизационог алгоритма



Минимизација вероватноће проналажења истих решења

- Проценом положаја локалних минимума на основу свих елемената скупа генерисаног у првом кораку, или на основу неког његовог подскупа
- Остварује се минимална вероватноћа вишеструких претраживања истих делова оптимизационог простора
- Ова особина је од кључног значаја за ефикасност алгоритама ове класе

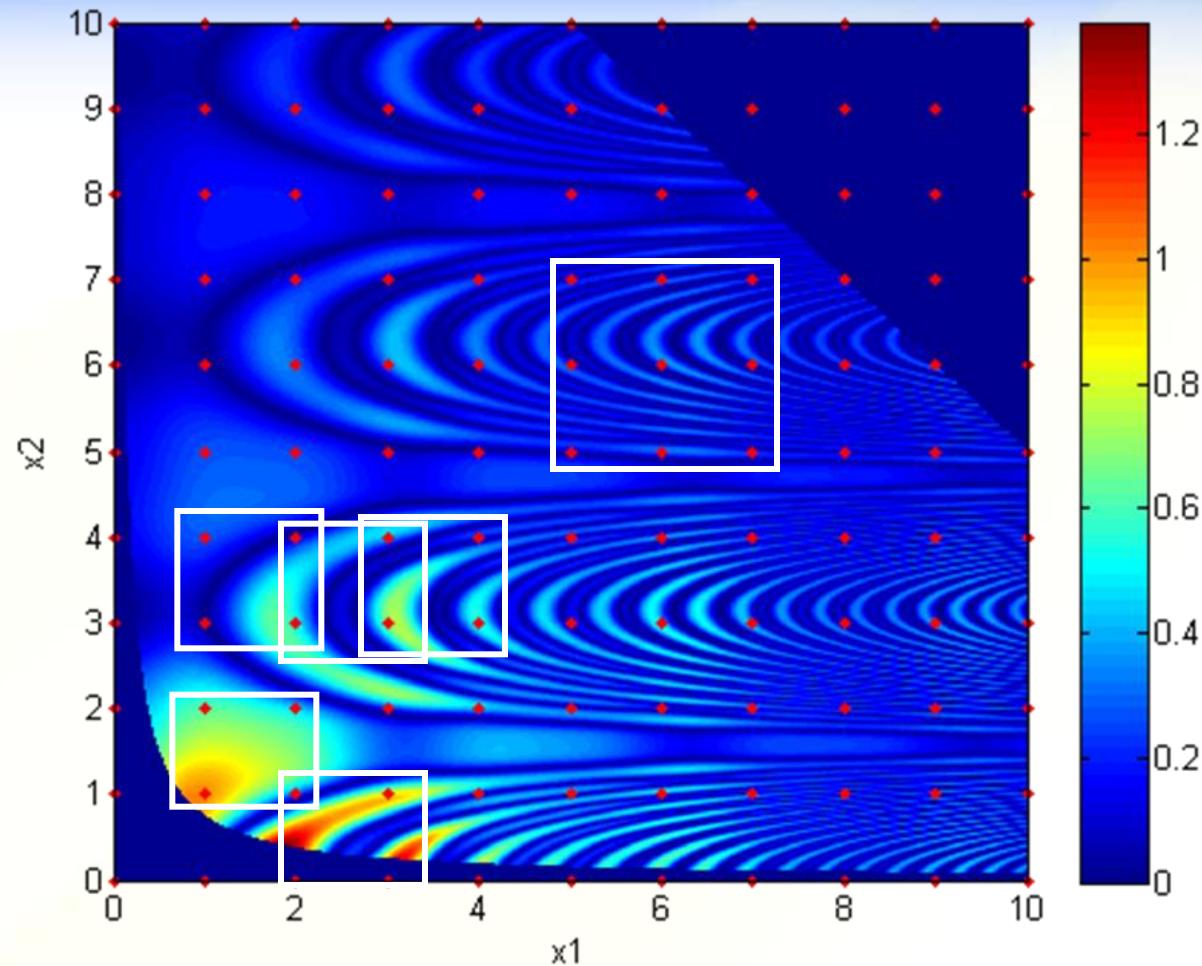
Систематско и случајно претраживање као први корак

- Систематско и случајно претраживање су природни кандидати за први корак алгоритма
- Оба алгоритма генеришу тачке по читавом оптимизационом простору
- Систематско претраживање има предност само за просторе са малим бројем димензија, грубо, до највише три димензије
- Код простора са већим бројем димензија овакав приступ генерисању тачака није погодан због огромног броја потребних итерација
- Стога, случајно претраживање са унiformном расподелом остаје као **једини практичан избор** за вишедимензионалне просторе

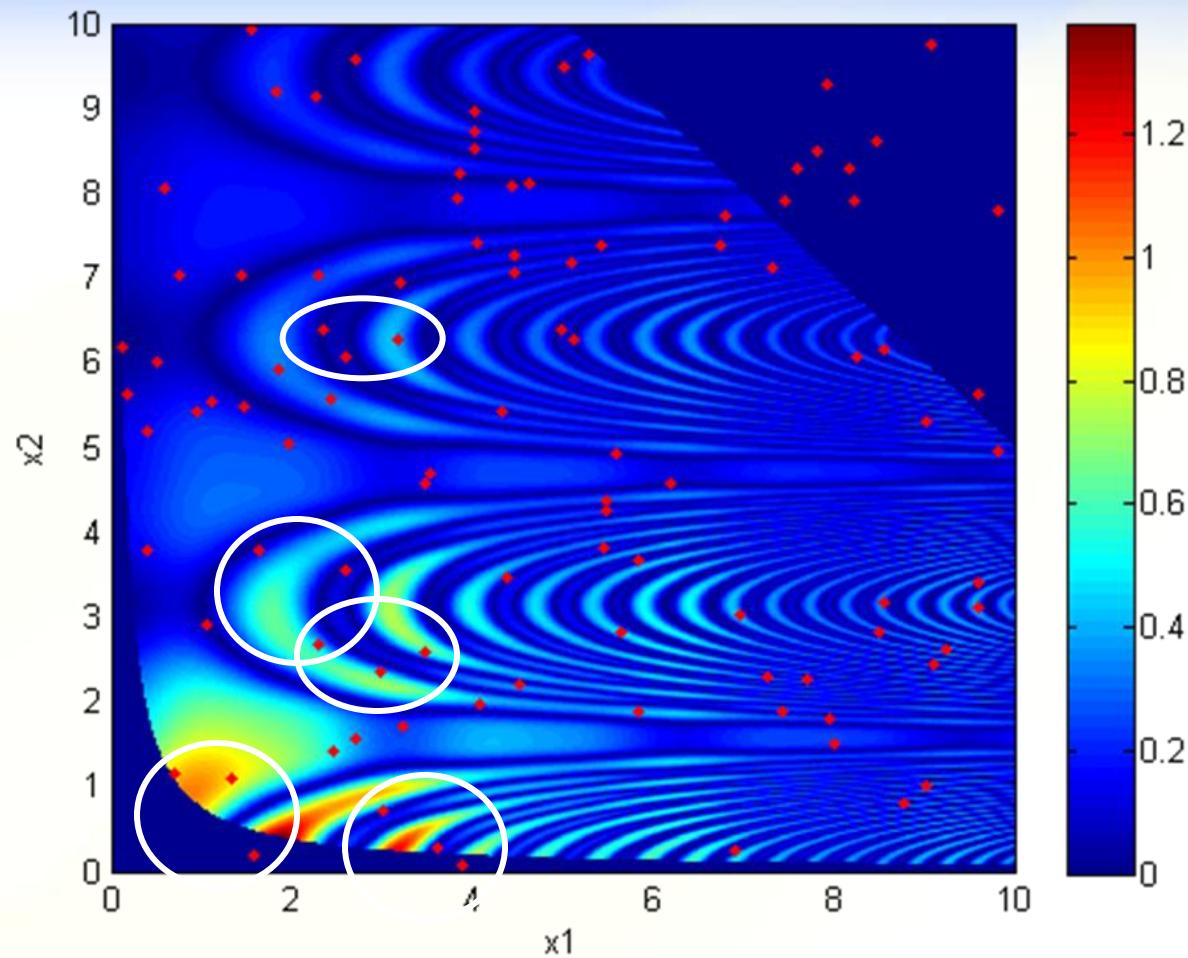
Процена локалних минимума

- Проценом локалних минимума пожељно је пронаћи околине свих минимума са најмањим могућим бројем рачунских операција
- Процена се може извршити на много различитих начина
- Један једноставан начин да се то уради јесте да се за сваку тачку из основног скупа пронађе M најближих тачака и уколико је вредност функције грешке у тој тачки мања од свих M суседних, тачка се прогласи локалним минимумом

Процена локалних минимума систематским претраживањем



Процена локалних минимума случајним претраживањем



Параметри процене локалних минимума

- Број елемената скупа случајних решења N је први параметар овако организованог алгоритма
- Потребно је да буде довољно велик да је могућа процена свих, или бар већине локалних минимума
- Овај број зависи од конкретног проблема оптимизације
- Експериментално је утврђено да би број суседних тачака M за процену локалних минимума требало да буде бар два пута већи од броја димензија простора
- За добро изабране параметре N и M број процењених локалних минимума конвергира стварном броју локалних минимума
- Сложеност обраде $O(N^2)$, у најгорем случају

Поређење метода на аналитички задатим функцијама

- Циљ: сагледати особине и утицај параметара алгоритама
- Стохастички алгоритми:
результат је **случајна променљива**
 - посматрамо средњу вредност најбољег решења после одговарајућег броја итерација
 - може се посматрати и девијација средњег најбољег решења

Сума квадрата: градијентни метод, константан корак

Број димензија	Дужина корака				
	10^{-1}	10^{-2}	10^{-3}	10^{-4}	10^{-5}
1	9	32	257	2507	25007
2	17	44	367	3549	35364
3	15	54	444	4341	43312
4	18	63	513	5013	50013
5	28	71	574	5605	55924
10	33	104	828	7943	79082
15	62	132	1003	9717	96877
20	56	157	1163	11225	111848

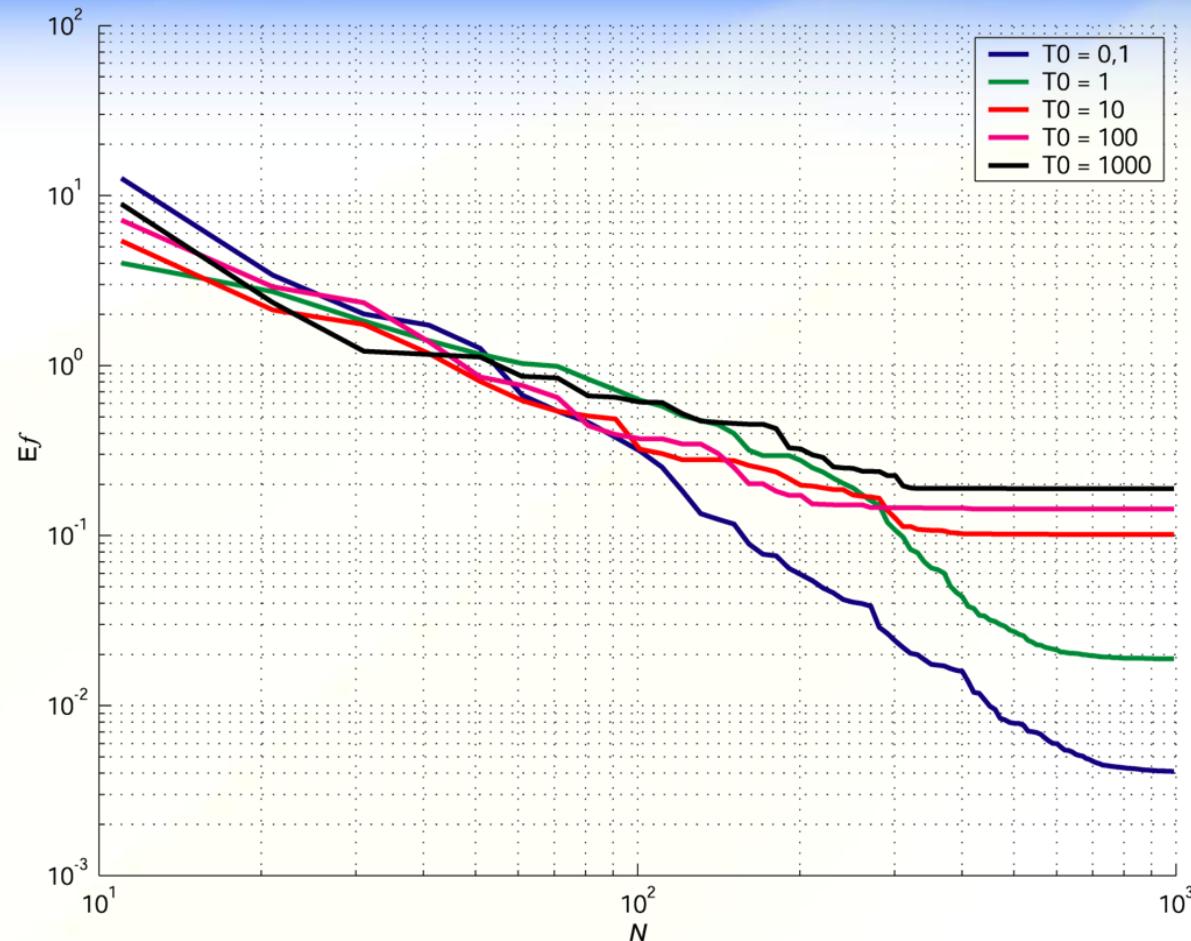
Сума квадрата: градијентни метод, удвојујуће корака

Дим.	Дужина почетног корака									
	10^{-1}	10^{-2}	10^{-3}	10^{-4}	10^{-5}	10^{-6}	10^{-7}	10^{-8}	10^{-9}	10^{-10}
1	9	17	20	36	61	64	100	103	147	157
2	11	24	34	46	60	99	106	120	172	195
3	19	29	42	52	74	107	94	153	173	133
4	22	28	38	58	80	93	131	130	185	188
5	25	37	45	73	102	96	156	175	170	260
10	28	59	93	84	158	123	156	222	216	263
15	56	63	103	151	149	181	177	270	289	267
20	137	99	130	211	228	184	321	317	297	435

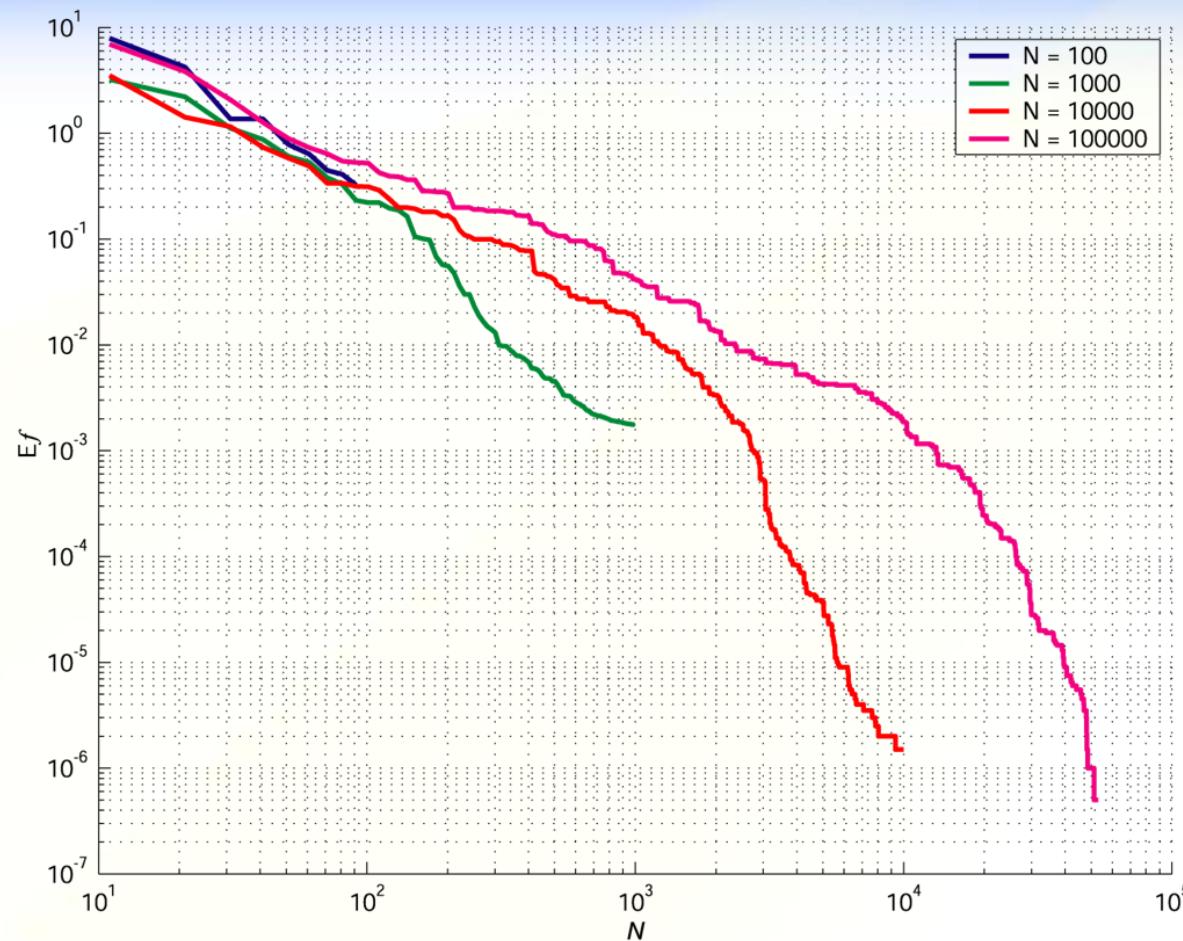
Сума квадрата: симплекс алгоритам

Дим.	Толеранција за функцију грешке									
	10^{-1}	10^{-2}	10^{-3}	10^{-4}	10^{-5}	10^{-6}	10^{-7}	10^{-8}	10^{-9}	10^{-10}
1	6	10	14	18	20	25	29	33	35	39
2	18	26	33	37	45	53	57	68	72	78
3	33	41	52	62	80	89	102	114	122	130
4	40	63	74	90	104	117	133	152	166	187
5	54	63	99	123	159	183	202	241	289	317
10	115	167	209	260	302	345	379	433	466	532
15	203	289	332	411	458	533	600	664	733	804
20	338	459	558	663	787	928	1102	1289	1428	1623

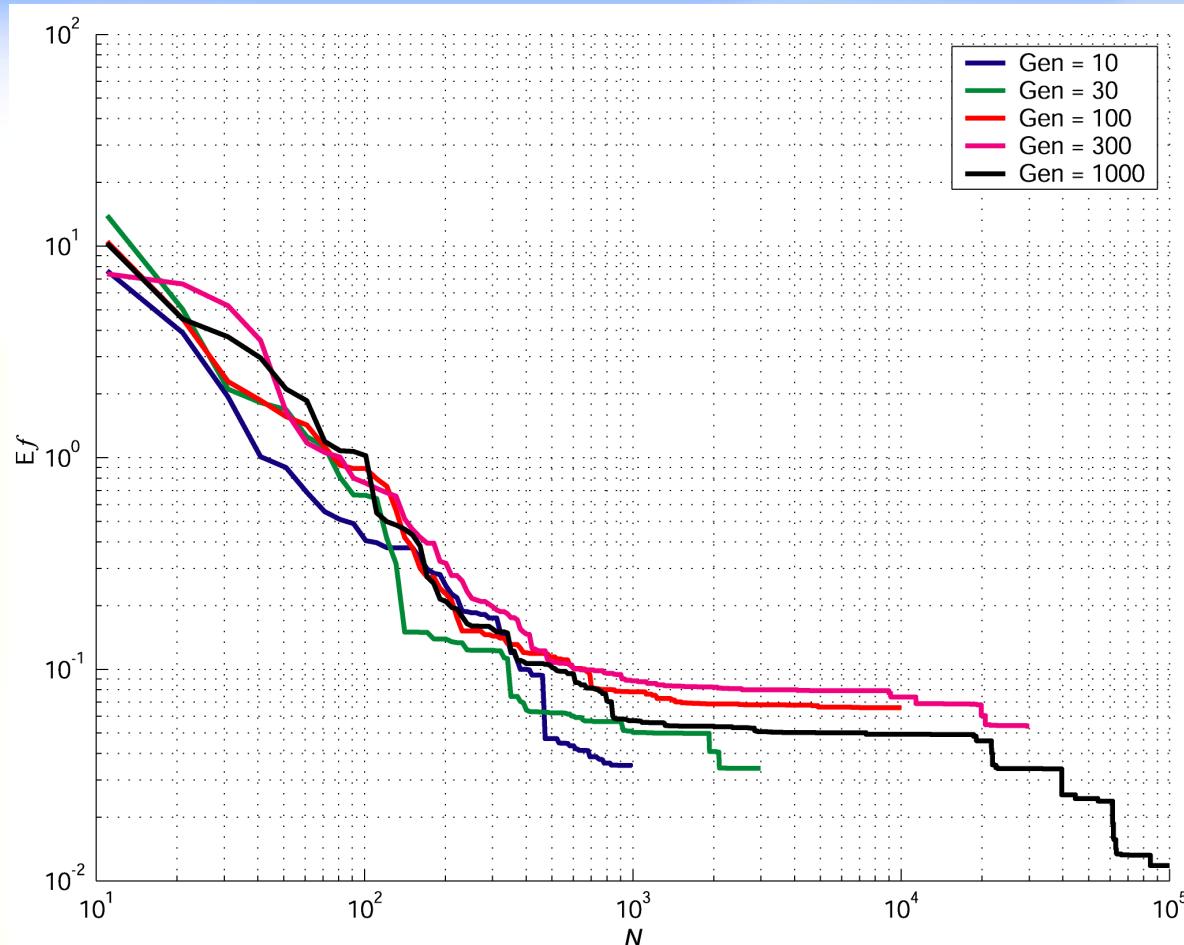
2D Розенброкова функција: СА – почетна температура



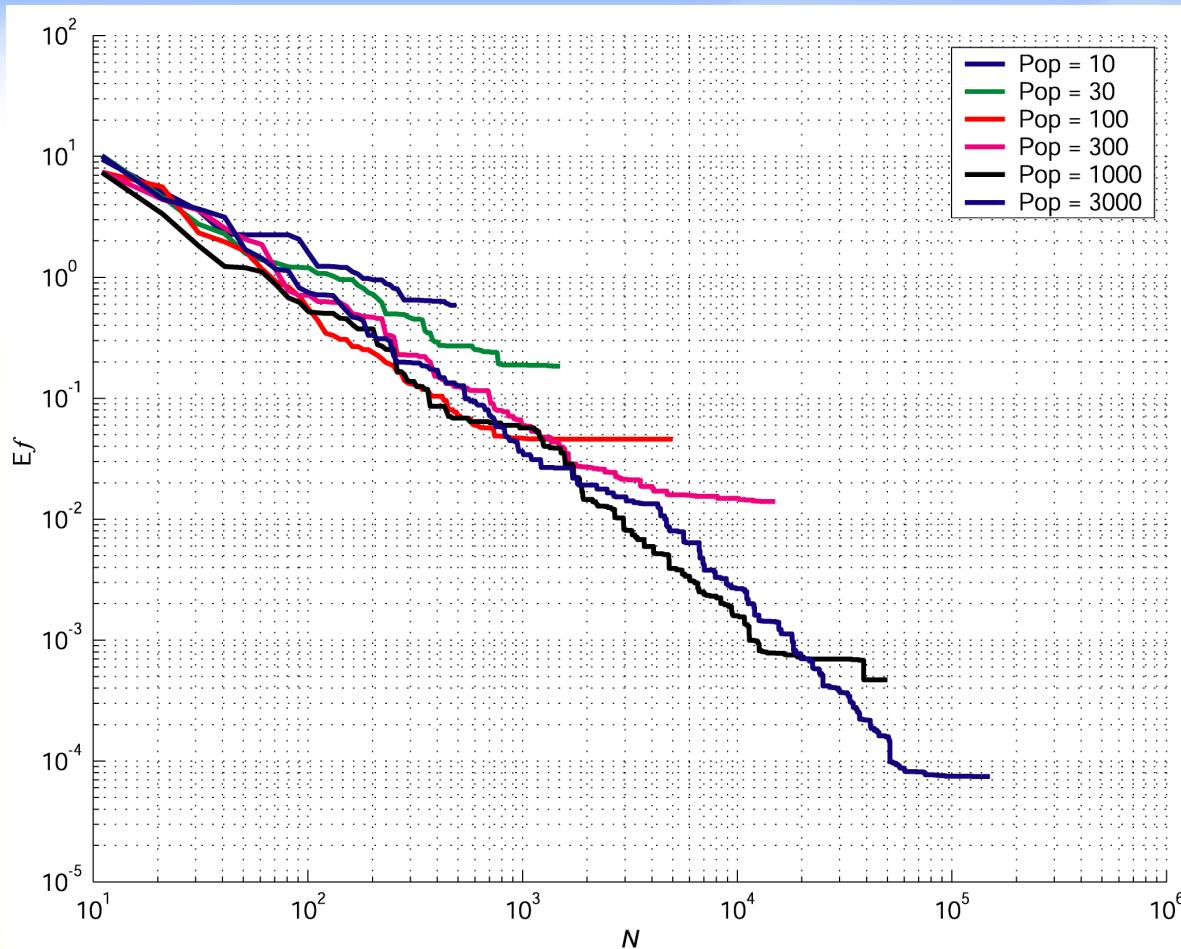
2D Розенброкова функција: СА – број итерација



2D Розенброкова функција: ГА – број генерација



2D Розенброкова функција: ГА – величина популације



Шекелова функција: алгоритми засновани на понављању

- **Рандом(1)+симплекс(100).** Симплекс алгоритам са максимално 100 итерација је стартован из случајно изабране тачке у оптимизационом простору.
- **Рандом(50)+симплекс(50).** Симплекс алгоритам са максимално 50 итерација је стартован из најбоље тачке из скупа од 50 случајно изабраних тачака.
- **ГА(50)+симплекс(50).** Полазна тачка за симплекс алгоритам са највише 50 итерација је проналажена генетичким алгоритмом. Параметри кориштеног генетичког алгоритма су: популација од 10 решења, 3 најбоља решења из генерације формирају наредну генерацију, укупно има 5 генерација, вероватноћа мутације $P_{mut} = 0,1$, вероватноћа укрштања $P_{cross} = 0,8$ и ниједно решење из претходне генерације не прелази у наредну. Прва генерација је бирана на случајан начин.
- **ПЛМ – процена локалних минимума.** Симплекс алгоритам са највише 50 итерација је стартован из свих процењених локалних минимума. Процена је вршена из најближих $M = 8$ тачака.

Шекелова 2D функција са једнаким минимумима (9 мин.)

Алгоритам	Укупан број итерација				
	500	1000	1500	2000	2500
Рандом(1) + симплекс(100)	3,6	6,0	7,8	8,0	8,4
Рандом(50) + симплекс(50)	4,2	6,2	7,8	8,6	9,0
ГА(50) + симплекс(50)	4,0	7,4	8,2	8,4	9,0
ПЛМ	7,8	9,0	9,0	9,0	9,0

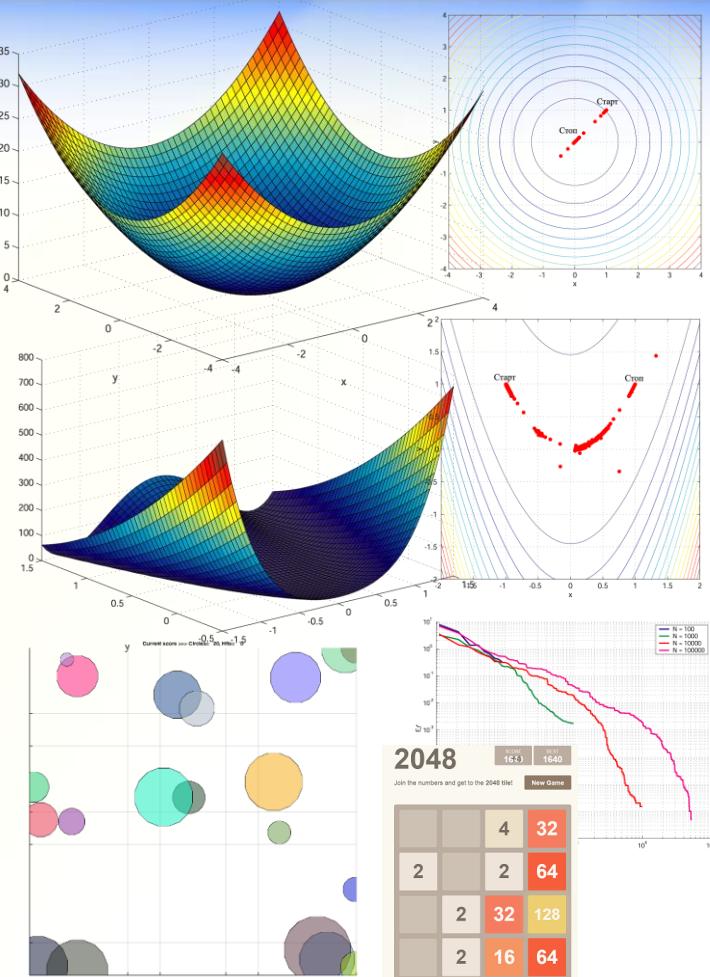
Шекелова 2D функција са неједнаким минимумима (9 мин.)

Алгоритам	Укупан број итерација				
	500	1000	1500	2000	2500
Рандом(1) + симплекс(100)	4,3	6,1	6,8	7,9	8,2
Рандом(50) + симплекс(50)	3,8	4,9	6,1	6,6	6,9
ГА(50) + симплекс(50)	4,0	5,5	6,7	6,8	6,7
ПЛМ	8,3	9,0	9,0	9,0	9,0

Инжењерске оптимизације

- поређење алгоритама -

- Оптимизације комплексних описних функција са више димензија
 - изабрати један оптимизациони алгоритам са курса
 - тест оптимизационе функције http://infinity77.net/global_optimization/test_functions.html#test-functions-index
 - сумирати резултате понашања алгоритма за различит број димензија оптимизационог проблема и различите параметре алгоритма
- Програмирање једноставних рачунарских игара
 - поналажење пута на шаховској табли
 - управљање једноставним играма коришћењем оптимизационих алгоритама



Инжењерске оптимизације - такмичарски проблеми -

The screenshot shows a web browser window with the URL codingcompetitions.withgoogle.com/hashcode/archive. The page displays past problem statements for three competitions: 2016, 2017, and 2018. Each competition section contains two problems: 'Final Round' and 'Qualification Round'. The 2016 section has two problems: 'City Plan' and 'Router placement'. The 2017 section has two problems: 'Self-driving rides' and 'Streaming videos'. The 2018 section has two problems: 'Final Round' and 'Qualification Round'. Each problem statement includes a 'View problem' and 'Download file' link.

Competition	Final Round Problem	Qualification Round Problem
2016	City Plan	Router placement
2017	Self-driving rides	Streaming videos
2018	Final Round	Qualification Round

IEEE WORLD CONGRESS ON
COMPUTATIONAL INTELLIGENCE
(WCCI)

[http://www.gecad.isep.ipp.pt/
WCCI2018-SG-COMPETITION/](http://www.gecad.isep.ipp.pt/WCCI2018-SG-COMPETITION/)

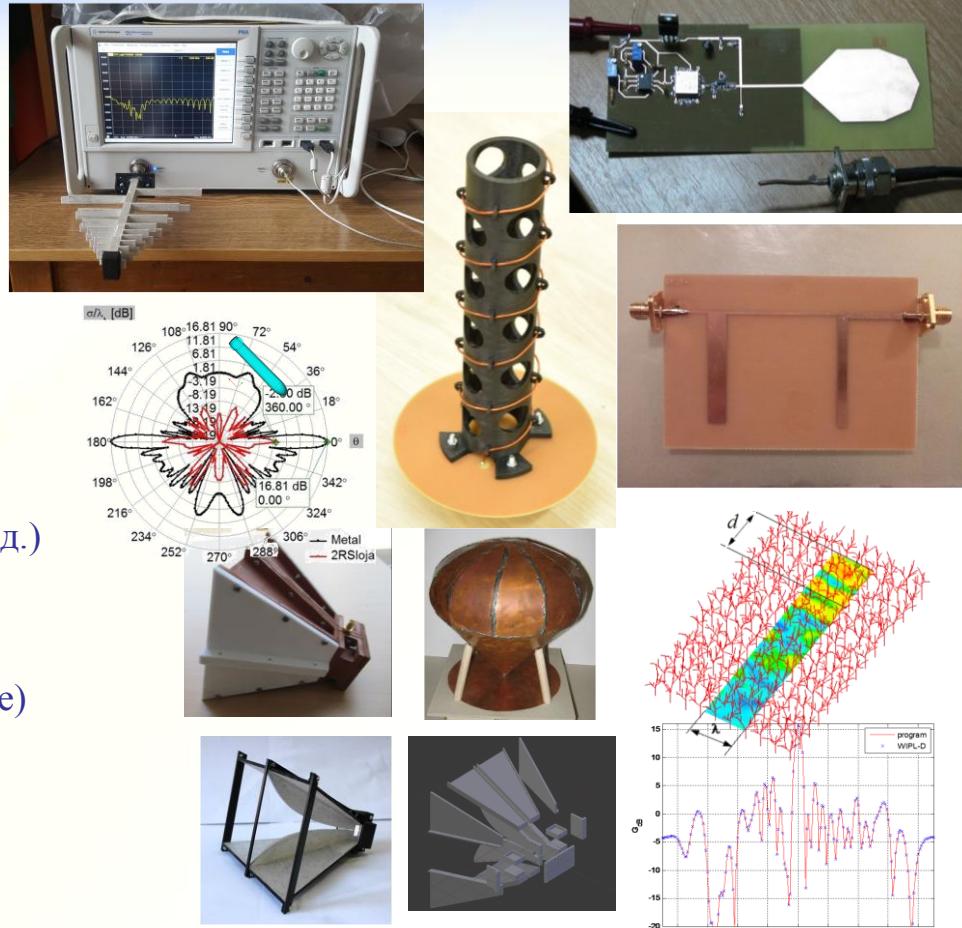
[http://www.gecad.isep.ipp.pt/
ERM-competitions/2020-2/](http://www.gecad.isep.ipp.pt/ERM-competitions/2020-2/)

<https://codingcompetitions.withgoogle.com/hashcode/archive>

Инжењерске оптимизације

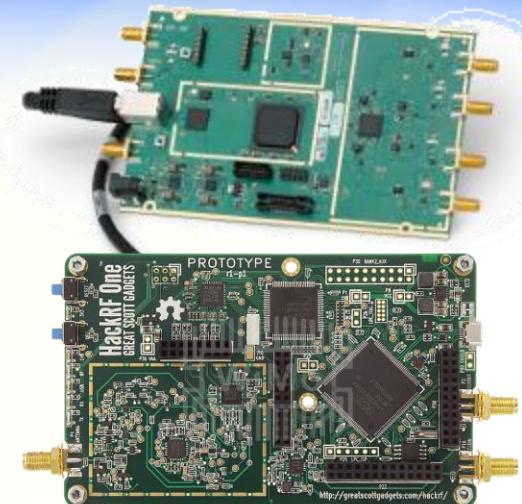
- примењена електромагнетика -

- Оптимизације антена и низова
 - Микротракасте антене
 - Низ микротракастих антена
 - Лог-периодична антена
 - Јаги антена
 - Хеликоидална антена
 - Вивалди антена
 - Широкопојасни монопол
 - Левак антена,...
- Оптимизација микроталасних кола
 - микроталасни филтри
(микротракасти, тракасти, таласоводни, итд.)
 - кола за прилагођење
(ускопојасна или широкопојасна)
 - трансформатори импеданси
(парето фронт вода променљиве импедансе)
- Оптимизација израчунавања нумеричких ЕМ језгара за симулације
- Оптимизовати, направити и премерити
(теме за радове)



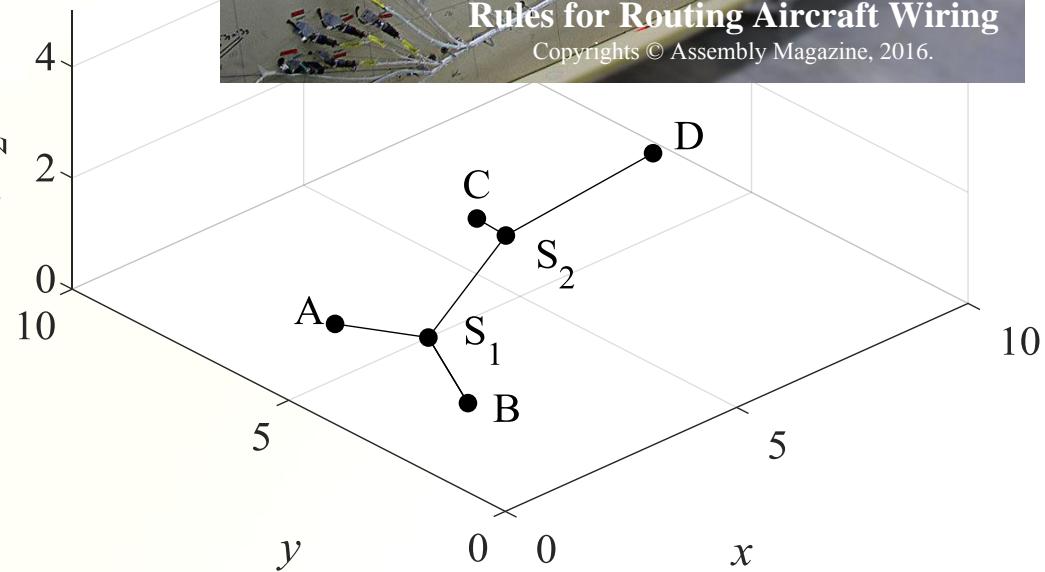
Инжењерске оптимизације - ЕМС, системи, електроника -

- Оптимизација сигнала генератора шума у опсезима WiFi, GPS, GSM, LTE коришћењем SDR
- Оптимално препознавање на основу спектра израченог ЕМ поља снимљеног помоћу SDR или NARDA SRM3006
- Анализа аналогних нелинеарних кола коришћењем оптимизационих алгоритама
 - кола са више диода (усмерачи и исправљачи)
 - кола са транзисторима (појачавачи итд.)
- Оптимизације управљања хардвером и софтвером

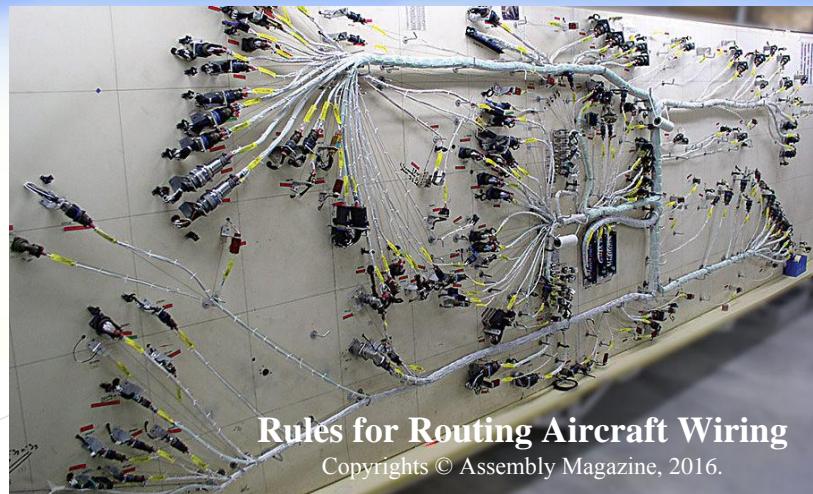


Задатак за вежбе: поставка

- Дате су четири тачке у Декартовом координатном систему:
- $A(1,5,1)$, $B(3,2,0)$, $C(5,7,1)$ и $D(6,3,3)$
- Потребно их је повезати праволинијским путевима тако да **укупна дужина пута буде минимална**



*** Steiner tree problem



Задатак за вежбе: детаљи

- Тачке се повезују додавањем двеју тачака $S_1(x_1, y_1, z_1)$ и $S_2(x_2, y_2, z_2)$
- Тачке A и B повезују се прво са S_1 , а тачке C и D прво са S_2 , као што је приказано на слици
- Одредити и записати у ASCII фајл
 - оптималне координате тачака S_1 и S_2 и ;
 - израчунати минималну дужину пута
- Имплементирати алгоритам за **оптимизацију јатом (PSO)** и искористити га за проналажење оптималних координата

