Multidiszciplináris tudományok, 11. kötet. (2021) 4 sz. pp. 111-119 **https://doi.org/10.35925/j.multi.2021.4.13**

KLASZTEREZÉS SZIMULÁLT LEHŰTÉSSEL ÉS GENETIKUS ALGORITMUSSAL

Agárdi Anita
tanársegéd, Miskolci Egyetem, Informatika Intézet, Általános Informatikai Intézeti Tanszék
3515 Miskolc, Miskolc-Egyetemváros, e-mail: agardianita@iit.uni-miskolc.hu

Absztrakt

A cikk a klaszterezés témakörével foglalkozik. A klaszterezés olyan adatbányászati algoritmus, amely a bemeneti adatokat csoportosítja egymáshoz kapcsolódó hasonlóságuk alapján. A kutatásban a klaszterezést viszont nem a szokványos klasszikus módszerrel oldottam meg, hanem metaheurisztikákkal. A metaheurisztikák közül a szimulált lehűtés és a genetikus algoritmus lett kiválasztva. A cikkben három adatsorra mutatok futási eredményeket, melyek alapján megállapítható, hogy az algoritmusok egész szépen megtalálják a klaszterhatárokat.

Kulcsszavak: *klaszterezés, genetikus algoritmus, szimulált lehűtés*

Abstract

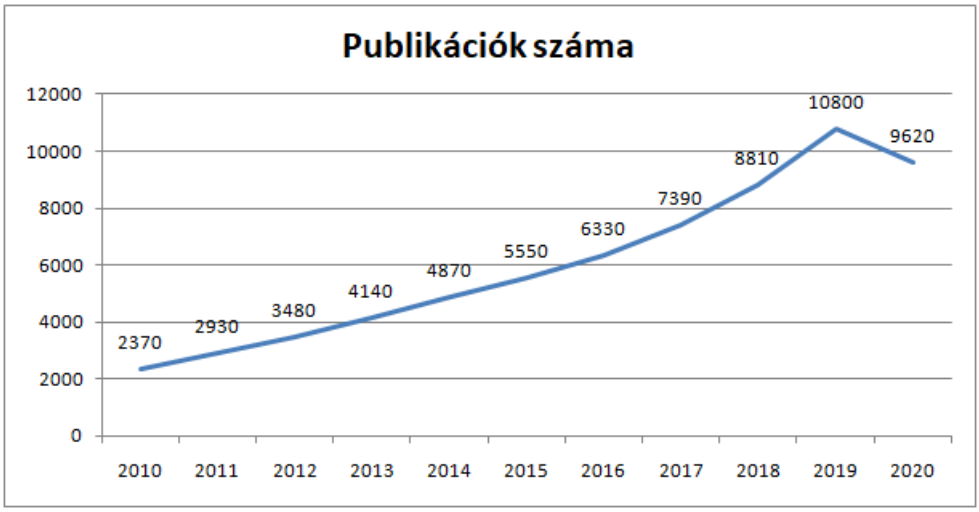
The article focuses on clustering algorithms. Clustering is a data mining algorithm that groups input data based on their related similarities. In the research, however, the clustering was not solved by the usual classical method, but by metaheuristics. From the metaheuristics, the simulated annealing and the genetic algorithm were selected. In this article, I present test results for three data sets, based on which it can be concluded that the algorithms find the cluster boundaries efficiently.

Keywords: *clustering, genetic algorithm, simulated annealing*

1. Bevezetés

A cikk a klaszterezés témakörével foglalkozik. A klaszterezés olyan adatbányászati algoritmus, amely során csoportok kialakítása a cél. A csoportokat úgy kell kialakítani, hogy az egymáshoz hasonló elemek azonos klaszterbe, míg a különböző elemek különböző klaszterbe kerüljenek. A klasszikus klaszterező algoritmusokon felül, mint a partíciós, hierarchikus, sűrűség alapú, rács alapú stb, a metaheurisztikák irányába is folynak kutatások. A metaheurisztikus algoritmusok számos feladatra adaptálhatóak, így akár klaszterezésre is. Ilyen algoritmusok például a genetikus algoritmus [1-2], szimulált lehűtés [3-5], részecske-raj alapú optimalizálás [6-7], hangya kolónia [8-9], tabu keresés [11-12], harmónia keresés [13-14], flower pollination [15-16], memetikus algoritmus [17] stb.

Az *1. ábra* a megjelent publikációkat évenkénti bontásban mutatja. A kereséshez a Google scholar adatbázist használtam, a keresés kulcsszavának pedig a következőt adtam meg: „metaheuristic clustering”. Látható, hogy 2010-2019-es időszakban folyamatosan növekedett a témában megjelent cikkek száma.



1. ábra. *Publikációk évenkénti bontásban (Google scholar adabázis „metaheuristic clustering" keresséssel)*

Jelen cikkben a metaheuriszikák közül kettőt választottam ki, ez a genetikus algoritmus és a szimulált lehűtés. A genetikus algoritmus [1-2] a természetben lezajló evolúciós folyamatokat modellezi. Egy kezdeti populáció generálás után keresztezési és mutációs lépésekkel új egyedek keletkeznek. Az új egyedek egy bizonyos (legrátermettebb) része lesz a következő populáció eleme. A genetikus algoritmus során az alábbi keresztezési operátorokat használtam: sorrendi keresztezés, részleges megfeleltetésű keresztezés, ciklikus keresztezés. Mutációnak az élcserét alkalmaztam, ami azt jelenti, hogy egy szekvencia elemeit megfordítjuk. A szimulált lehűtés [3-5] során egyetlen megoldásból indulunk ki. Kezdetben ez az aktuális megoldás. Az aktuális megoldás egy szomszédját vizsgáljuk. Ha jobb a szomszéd, mint az aktuális megoldás, akkor elfogadjuk. Ha nem jobb, akkor is egy bizonyos valószínűséggel elfogadjuk. A rosszabb megoldás elfogadásának valószínűsége az iterációval csökken. Mivel az algoritmus a rosszabb megoldásokat is képes elfogadni, így könnyedén kijut egy lokális optimumból. Itt szintén az élcsere operátort használtam szomszédság keresésre. A cikk következő fejezetében a futási eredményeket mutatom be.

2. Futási eredmények

Ezen fejezet a genetikus algoritmus és a szimulált lehűtés algoritmus eredményeit mutatja be. Három adatsort készítettem, melyek az alábbi paraméterekkel rendelkeznek: pontok száma (100, 50, 90), klaszterek optimális száma (10, 5, 3). Az adatsorok jól klaszterezhetőek, mert egyes pontok egymáshoz nagyon közel, míg más pontok nagyon távol találhatóak. A klaszterezés elvált eredménye tehát az, hogy az első adatsort 10, a második adatsort 5, a harmadik adatsort pedig 3 csoportra bontsa. Az elvárás tovább az, hogy az egyes klaszterek szemmel láthatóan is távol legyenek egymástól. A klaszterezés során permutációs ábrázolásmódot használtam. Ez azt jelenti, hogy a permutáció elemei az adatsor egyes elemeit jelenti. A kiértékelés során a permutáció egyes elemeit veszem sorban addig, amíg elég közel helyezkednek el egymáshoz (az adatsor elemei közötti távolságok átlagának negyedik gyöke). Ezek azonos klaszterbe fognak kerülni. Amennyiben a permutáció egy következő eleme már távol helyezkedik el az aktuális elemhez képest, akkor az elem már egy másik klaszterbe fog kerülni. A klaszterezés jóságát is mérni szükséges, hogy az iterációk során meg lehessen különböztetni, hogy

melyik klaszterezés (permutáció) jelent jó klaszterezést. Erre az egyes klaszterbeli elemek egymáshoz képesti távolságát használom. Az egyes klaszterbeli elemek egymástól vett távolságainak összege lesz a fitnesz érték. Ennek az értéknek a minimalizálása a cél. Az adatsor és a klaszterezés kialakítási stratégia a [18] cikkben bemutatottak alapján készült.

Az algoritmusok nem véletlenszerű megoldásból indulnak ki, hanem egy olyan permutációból (pontok sorrendjéből), amelyet úgy kapunk, hogy körutat képzünk a véletlen pont beszúrása algoritmussal. Az algoritmus során kezdetben egyetlen pontot tartalmaz a körút, majd folyamatosan szúrjuk be az egyes pontokat a körútba arra a helyre, ahol a beszúrás költsége (körút növekedése) minimális [10].

Elsőként az algoritmusok paraméterezését mutatom be, ezután a felhasznált adatsort. Majd a kialakult klaszterezésről ábrákat, és a futási eredményeket táblázatos formában (klaszterek száma, fitnesz érték, futási idő). A felhasznált adatsorok struktúráját a függelék F1 táblázata tartalmazza.

Az *1. táblázat* mutatja a Genetikus algoritmus paraméterezését. Iterációszámnak egy nagy értéket adtam, a keresési tér hatékony felkutatása végett. A populációszámnak szintén nagy értéket adtam. Az elitizmus azt jelenti, hogy a populáció bizonyos legjobb egyedei változatlanul kerülnek a következő populációba. Itt egy alacsony számot érdemes megadni. A keresztezésnek egy nagy valószínűséget szokás megadni az algoritmus tervezése során. Az algoritmus 90% valószínűséggel keresztez összességében (30% valószínűséggel sorrendi, 30% valószínűséggel részleges megfeleltetésű és 30% valószínűséggel ciklikus keresztezést hajt végre). Mutációnak a keresztezéstől sokkal kisebb értéket szokás adni, én 30%-ot választottam.

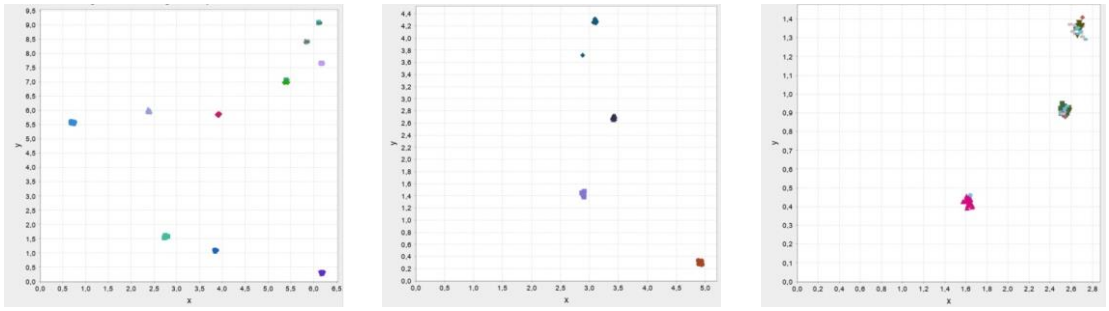
1. táblázat. *Genetikus algoritmus paraméterei*

Paraméter	**Érték**
Iterációszám	100
Populációszám	60
Elitizmusszám	10
Sorrendi keresztezési ráta	0.3
Részleges megfeleltetésű keresztezés ráta	0.3
Ciklikus keresztezés ráta	0.3
Mutációs ráta	0.3

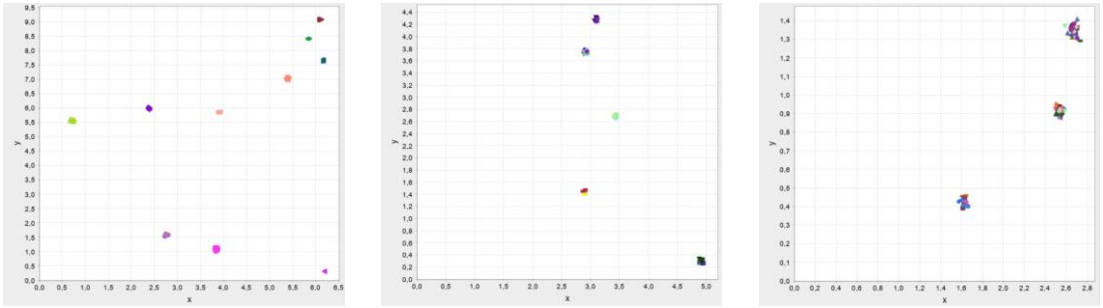
A szimulált lehűtés algoritmus paraméterezését a *2. táblázat* mutatja. Itt is az iterációszámnak egy nagy értéket adunk a keresési tér hatékony feltérképezése végett. Az alfa érték a hőmérséklet csökkenését jelzi, 15%-al csökken az egyes iterációknál a hőmérséklet. A hőmérsékletnek egy nagy értéket szokás adni, ez azt jelzi, hogy mennyi az esélye annak, hogy a szomszéd megoldást (amely rosszabb) elfogadjuk. A hossznak egy kis számot szokás megadni, ez azt jelzi, hogy hány iterációnál maradjon változatlan a hőmérséklet. Jelen példában minden 20. iterációnál csökken a hőmérséklet 15%-al.

2. táblázat. *Szimulált lehűtés algoritmus paraméterei*

Paraméter	**Érték**
Iterációszám	120
Alfa	0.85
Hőmérséklet	1000
Hossz	20



2. ábra. *Genetikus algoritmus (Genetic algorithm) eredménye az első, második és harmadik adatsorra*



3. ábra. *Szimulált lehűtés algoritmus (Simulated Annealing algorithm) eredménye az első, második és harmadik adatsorra*

A futási eredmények alapján a genetikus algoritmus az első két adatsorra hatékonyan megoldotta a feladatot. A harmadik adatsort viszont nem a várt módon klaszterezte. A szimulált lehűtés csak az első adatsort klaszterezte hatékonyan, a második és a harmadik adatsornál viszont nem találta meg a klaszterhatárokat, a vártnál sokkal több klaszter alakult ki. A futási eredményeket táblázatba is foglaltam, ahol a klaszterszámot, fitness értéket és a futási időt is feltüntettem.

3. táblázat. *Futási eredmények az első adatsorra*

Algoritmus	Klaszterek száma	Fitnesz érték	Futási idő (perc)
Genetic Algorithm	17	0.6583	0.0096
Simulated Annealing	10	0.3459	$9.3251*10^{-4}$

Az első adatsort a szimulált lehűtésnek sikerült 10 csoportra bontani, de a genetikus algoritmus eredménye ettől jócskán távol maradt (*3. táblázat*).

4. táblázat. *Futási eredmények a második adatsorra*

Algoritmus	Klaszterek száma	Fitnesz érték	Futási idő (perc)
Genetic Algorithm	4	0.6128	0.0035
Simulated Annealing	17	0.4082	$9.9647*10^{-4}$

A második adatsornál 5 klaszter lett volna az optimális, a genetikus algoritmus ezt majdnem elérte. A szimulált lehűtés ettől nagyon távol maradt, 17 csoportra bontotta az adatsort (*4. táblázat*).

5. táblázat. *Futási eredmények a harmadik adatsorra*

Algoritmus	Klaszterek száma	Fitnesz érték	Futási idő (perc)
Genetic Algorithm	7	1.1570	0.0052
Simulated Annealing	28	0.7223	$7.2960*10^{-4}$

A harmadik adatsornál 3 csoport lett volna az optimális, mindkét algoritmus ettől sokkal több klaszterre bontotta a pontokat, a genetikus algoritmus 7, a szimulált lehűtés pedig 28 csoportot alakított ki.
A futási idők mindhárom adatsor esetén alacsonyak voltak (*5. táblázat*).

3. Összegzés

Jelen cikkben a klaszterezés, mint az egyik leggyakoribb adatbányászati feladat került középpontba. A cikkben két metaheurisztika került bemutatásra, melyet klaszterezésre használtam. Ez a genetikus algoritmus és a szimulált lehűtés. A cikkben három, jól klaszterezhető adatsorra tárgyaltam az algoritmusok hatékonyságát. A futási eredmények alapján az algoritmusok az első két adatsort viszonylag jól csoportosították, míg a harmadik adatsornál sokkal több csoportra bontották a pontokat. A szimulált lehűtés az első adatsornál szép eredményt adott, de a második és harmadik adatsornál nem találta meg az optimális klaszterezést.

Köszönetnyilvánítás

„A cikkben/előadásban/tanulmányban ismertetett kutató munka az EFOP-3.6.1-16-2016-00011 jelű „Fiatalodó és Megújuló Egyetem – Innovatív Tudásváros – a Miskolci Egyetem intelligens szakosodást szolgáló intézményi fejlesztése" projekt részeként – a Széchenyi 2020 keretében – az Európai Unió támogatásával, az Európai Szociális Alap társfinanszírozásával valósul meg."

Irodalom

[1] Hruschka, E. R., & Ebecken, N. F. (2003). A genetic algorithm for cluster analysis. Intelligent Data Analysis, 7(1), 15-25. **https://doi.org/10.3233/IDA-2003-7103**

[2] Maulik, U., & Bandyopadhyay, S. (2000). Genetic algorithm-based clustering technique. Pattern recognition, 33(9), 1455-1465. **https://doi.org/10.1016/S0031-3203(99)00137-5**

[3] Klein, R. W., & Dubes, R. C. (1989). Experiments in projection and clustering by simulated annealing. Pattern Recognition, 22(2), 213-220. **https://doi.org/10.1016/0031-3203(89)90067-8**

[4] Brown, D. E., & Huntley, C. L. (1992). A practical application of simulated annealing to clustering. Pattern recognition, 25(4), 401-412. **https://doi.org/10.1016/0031-3203(92)90088-Z**

[5] Selim, S. Z., & Alsultan, K. (1991). A simulated annealing algorithm for the clustering problem. Pattern recognition, 24(10), 1003-1008. **https://doi.org/10.1016/0031-3203(91)90097-O**

[6] Van der Merwe, D. W., & Engelbrecht, A. P. (2003, December). Data clustering using particle swarm optimization. In The 2003 Congress on Evolutionary Computation, 2003. CEC'03. (Vol. 1, pp. 215-220). IEEE.

[7] Li, C., & Yang, S. (2009, May). A clustering particle swarm optimizer for dynamic optimization. In 2009 IEEE congress on evolutionary computation (pp. 439-446). IEEE. **https://doi.org/10.1109/CEC.2009.4982979**

[8] Shelokar, P. S., Jayaraman, V. K., & Kulkarni, B. D. (2004). An ant colony approach for clustering. Analytica Chimica Acta, 509(2), 187-195. **https://doi.org/10.1016/j.aca.2003.12.032**

[9] İnkaya, T., Kayalıgil, S., & Özdemirel, N. E. (2015). Ant colony optimization based clustering methodology. Applied Soft Computing, 28, 301-311. **https://doi.org/10.1016/j.asoc.2014.11.060**

[10] Azar, Y. (1994). Lower Bounds for Insertion Methods for TSP. Combinatorics, Probability & Computing, 3, 285-292. **https://doi.org/10.1017/S096354830000119X**

[11] Lu, Y., Cao, B., Rego, C., & Glover, F. (2018). A Tabu Search based clustering algorithm and its parallel implementation on Spark. Applied Soft Computing, 63, 97-109. **https://doi.org/10.1016/j.asoc.2017.11.038**

[12] Li, W., Wang, G., & Li, K. (2017). Clustering algorithm for audio signals based on the sequential Psim matrix and Tabu Search. EURASIP Journal on Audio, Speech, and Music Processing, 2017(1), 26. **https://doi.org/10.1186/s13636-017-0123-3**

[13] Gupta, G. P., & Jha, S. (2018). Integrated clustering and routing protocol for wireless sensor networks using Cuckoo and Harmony Search based metaheuristic techniques. Engineering Applications of Artificial Intelligence, 68, 101-109. **https://doi.org/10.1016/j.engappai.2017.11.003**

[14] Moh'd Alia, O. (2018). A dynamic harmony search-based fuzzy clustering protocol for energy-efficient wireless sensor networks. Annals of Telecommunications, 73(5-6), 353-365. **https://doi.org/10.1007/s12243-017-0611-6**

[15] Kaur, A., Pal, S. K., & Singh, A. P. (2020). Hybridization of Chaos and Flower Pollination Algorithm over K-Means for data clustering. Applied Soft Computing, 97, 105523. **https://doi.org/10.1016/j.asoc.2019.105523**

[16] Dhal, K. G., Gálvez, J., & Das, S. (2019). Toward the modification of flower pollination algorithm in clustering-based image segmentation. Neural Computing and Applications, 1-19.

[17] Sheng, W., Chen, S., Sheng, M., Xiao, G., Mao, J., & Zheng, Y. (2016). Adaptive multisubpopulation competition and multiniche crowding-based memetic algorithm for automatic data clustering. IEEE transactions on evolutionary computation, 20(6), 838-858.

[18] Agárdi, A. (2021) Klasszikus klaszterező algoritmusok módosítása körút alapon, Multidiszciplináris Tudományok, 11(4), 81-86. **https://doi.org/10.35925/j.multi.2021.4.9**

Függelék

F1. táblázat. *Az adatsorok struktúrája*

Első adatsor			**Második adatsor**			**Harmadik adatsor**		
	Pontok száma	**Opt. klaszter-szám**		**Pontok száma**	**Opt. klaszter-szám**		**Pontok száma**	**Opt. klaszter-szám**
	100	10		50	5		90	3
	Koordináták			**Koordináták**			**Koordináták**	
Sor-szám	**X**	**Y**	**Sor-szám**	**X**	**Y**	**Sor-szám**	**X**	**Y**
1	6,1747	7,6551	1	2,8828	3,7170	1	2,6909	1,3597
2	6,1766	7,6399	2	2,8884	3,7239	2	2,7044	1,3668
3	6,1759	7,6532	3	2,9260	3,7745	3	2,6695	1,3826
4	6,1727	7,6522	4	2,8922	3,7484	4	2,6635	1,3672
5	6,1487	7,6524	5	2,8987	3,7862	5	2,6848	1,3669
6	6,1838	7,6685	6	2,9215	3,7523	6	2,7048	1,3063
7	6,1945	7,6502	7	2,9584	3,7495	7	2,6501	1,3483
8	6,1777	7,6753	8	2,9384	3,7541	8	2,6147	1,3337
9	6,1788	7,6600	9	2,9279	3,7812	9	2,7033	1,3613
10	6,1450	7,6291	10	2,9464	3,7241	10	2,6580	1,3118
11	5,8577	8,4049	11	3,4392	2,7148	11	2,6753	1,3541
12	5,8562	8,4080	12	3,4040	2,7042	12	2,7060	1,3686
13	5,8678	8,4164	13	3,4185	2,6924	13	2,5899	1,3716
14	5,8481	8,3926	14	3,4115	2,6881	14	2,6397	1,3220
15	5,8549	8,4092	15	3,4060	2,6796	15	2,7344	1,2923
16	5,8282	8,4076	16	3,4372	2,6760	16	2,6803	1,3813
17	5,8339	8,4038	17	3,4004	2,6932	17	2,6825	1,3486
18	5,8625	8,4144	18	3,4157	2,7131	18	2,6417	1,3677
19	5,8466	8,4294	19	3,3990	2,6932	19	2,6955	1,3661
20	5,8362	8,4264	20	3,4160	2,6564	20	2,6897	1,3266
21	2,7416	1,5521	21	3,1182	4,2759	21	2,6628	1,3535
22	2,7474	1,6199	22	3,1095	4,2815	22	2,6892	1,3356
23	2,7480	1,5527	23	3,0920	4,2906	23	2,6644	1,3225
24	2,7408	1,5840	24	3,1021	4,3071	24	2,6584	1,3723
25	2,7878	1,5817	25	3,1182	4,2645	25	2,6778	1,3417
26	2,7412	1,5510	26	3,1140	4,2731	26	2,6760	1,3564
27	2,7228	1,5298	27	3,0935	4,3149	27	2,6435	1,3670
28	2,7825	1,5871	28	3,1167	4,2763	28	2,6426	1,3531
29	2,7259	1,5414	29	3,1078	4,2461	29	2,6651	1,3677
30	2,8014	1,5797	30	3,0688	4,2591	30	2,7059	1,4088
31	2,3693	6,0356	31	2,8662	1,4501	31	2,5191	0,8973
32	2,4098	5,9884	32	2,9028	1,4467	32	2,5472	0,9295
33	2,3676	5,9702	33	2,8893	1,4224	33	2,5457	0,8895
34	2,3920	5,9900	34	2,9046	1,4510	34	2,5323	0,9056
35	2,3940	6,0026	35	2,8938	1,4118	35	2,5177	0,9005
36	2,3723	5,9780	36	2,9103	1,4746	36	2,5021	0,8991

37	2,4065	5,9722	37	2,9066	1,3809	37	2,5219	0,9337
38	2,3670	5,9665	38	2,8991	1,4097	38	2,5125	0,9106
39	2,3617	6,0109	39	2,8938	1,4346	39	2,5243	0,9284
40	2,4020	5,9420	40	2,8704	1,4476	40	2,5359	0,8883
41	6,1124	9,0662	41	4,9346	0,3284	41	2,5291	0,8817
42	6,1159	9,0881	42	4,9327	0,3169	42	2,4984	0,9241
43	6,0968	9,0715	43	4,9339	0,3053	43	2,5392	0,9256
44	6,1284	9,0684	44	4,9204	0,3215	44	2,5444	0,9059
45	6,1021	9,0772	45	4,9462	0,2683	45	2,5482	0,9248
46	6,1036	9,0856	46	4,9178	0,2724	46	2,5313	0,9135
47	6,0831	9,0809	47	4,8839	0,2756	47	2,5203	0,9341
48	6,1155	9,0635	48	4,9317	0,2674	48	2,5598	0,9353
49	6,1232	9,0971	49	4,9166	0,2888	49	2,5356	0,9366
50	6,1286	9,0879	50	4,8755	0,3340	50	2,5275	0,9090
51	6,1730	0,2722				51	2,5038	0,8974
52	6,1723	0,2721				52	2,5542	0,9075
53	6,2176	0,2776				53	2,5849	0,9140
54	6,1786	0,2746				54	2,5420	0,9164
55	6,1874	0,3247				55	2,5382	0,9230
56	6,1721	0,2855				56	2,5767	0,9287
57	6,1513	0,3033				57	2,5219	0,8947
58	6,1952	0,3011				58	2,5425	0,8788
59	6,1537	0,2582				59	2,5601	0,8985
60	6,1904	0,2779				60	2,5131	0,9503
61	3,8704	1,1179				61	1,6374	0,4241
62	3,8238	1,1261				62	1,6407	0,4343
63	3,8305	1,0743				63	1,6461	0,4376
64	3,8341	1,0982				64	1,5938	0,4320
65	3,8364	1,1377				65	1,6477	0,4165
66	3,8383	1,0193				66	1,6643	0,4019
67	3,8719	1,0776				67	1,6304	0,4431
68	3,8337	1,1343				68	1,6349	0,4156
69	3,8182	1,0947				69	1,6347	0,4233
70	3,8583	1,0800				70	1,6235	0,4089
71	0,7362	5,5913				71	1,6413	0,4596
72	0,6905	5,5682				72	1,6103	0,4289
73	0,7203	5,5412				73	1,6227	0,4262
74	0,7217	5,5878				74	1,6069	0,4551
75	0,7637	5,5910				75	1,6117	0,4387
76	0,7172	5,5837				76	1,6296	0,4232
77	0,6774	5,5477				77	1,6118	0,3922
78	0,7192	5,6092				78	1,6356	0,4160
79	0,7490	5,5281				79	1,5738	0,4266
80	0,6756	5,6200				80	1,6279	0,4321
81	5,3902	7,0243				81	1,6582	0,3973
82	5,4321	6,9798				82	1,6252	0,4437
83	5,4190	6,9831				83	1,6185	0,4062
84	5,3863	7,0288				84	1,6305	0,4404

85	5,3874	6,9904				85	1,6249	0,4255
86	5,3554	6,9716				86	1,6186	0,4294
87	5,3962	7,0566				87	1,6089	0,4371
88	5,3648	6,9889				88	1,6363	0,4166
89	5,4255	7,0144				89	1,6335	0,4262
90	5,3957	7,0658				90	1,6224	0,4301
91	3,9045	5,8494						
92	3,9228	5,8704						
93	3,9092	5,8909						
94	3,9401	5,8782						
95	3,9134	5,8162						
96	3,9098	5,8776						
97	3,9243	5,8561						
98	3,8827	5,8519						
99	3,9018	5,8409						
100	3,9261	5,8503						