155ADKG: Konvexní obálky a jejich konstrukce

Datum odevzdání: 26.11.2017

Petra Millarová, Oleksiy Maybrodskyy

Contents

1 Zadání **3**

2 Popis a rozbor problému **4**

3 Popisy algoritmů **5**
3.1 Jarvis Scan 5
3.2 Quick Hull 6
3.3 Incremental Construcion 6
3.4 Graham Scan 6
3.5 Striktně konvexní obálka 6

4 Vstupní data **7**

5 Výstupní data **8**

6 Ukázky aplikace **9**
6.1 RANDOM 12
6.2 GRID 16
6.3 CLUSTER 20

7 Závěr **24**
7.1 Náměty na vylepšení 24

References **24**

1 Zadání

Následuje kopie oficiálního zadání úlohy. Autoři z nepovinných bodů zadání úspěšně implementovali algoritmus pro ošetření singulárního případu u Jarvis Scan: existence kolineárních bodů v data, rozpracovana kontrukce konvexní obálky metodou Graham Scan.

Úloha č. 2: Konvexní obálky a jejich konstrukce

Vstup: množina $P = \{p_1, ..., p_n\}$, $p_i = [x, y_i]$.

Výstup: $\mathcal{H}(P)$.

Nad množinou P implementujete následující algoritmy pro konstrukci $\mathcal{H}(P)$:

- Jarvis Scan.
- Quick Hull.
- Incremental Construction.

Vstupní množiny bodů včetně vygenerovaných konvexních obálek vhodně vizualizujte. Grafické rozhraní realizujte s využitím frameworku QT. Dynamické datové struktury implementujte s využitím knihovny STL.

Pro množiny $n \in < 1000, 1000000 >$ vytvořte grafy ilustrující doby běhu algoritmů pro zvolená n. Měření proveďte pro různé typy vstupních množin (náhodná množina, rastr, clustrovaná data) opakovaně (10x) a různá n (celkem 10) s uvedením rozptylu. Naměřené údaje uspořádejte do přehledných tabulek.

Zamyslete se nad problematikou možných singularit pro různé typy vstupních množin a možnými optimalizacemi. Zhodnoťte dosažené výsledky. Rozhodněte, která z těchto metod je s ohledem na časovou složitost a typ vstupní množiny P nejvhodnější.

Hodnocení:

Krok	**Hodnocení**
Konstrukce konvexních obálek metodami Jarvis Scan, Quick Hull, Incremental Construction.	15b
Konstrukce konvexní obálky metodou Graham Scan	*+5b*
Konstrukce striktně konvexních obálek pro všechny uvedené algoritmy.	*+5b*
Ošetření singulárního případu u Jarvis Scan: existence kolineárních bodů v datasetu.	*+2b*
Konstrukce Minimum area enclosing box některou z metod.	*+5b*
Algoritmus pro automatické generování konvexních/nekonvexních množin bodů různých tvarů (kruh, elipsa, čtverec, hvězda, popř. další).	*+4b*
Max celkem:	**36b**

Čas zpracování: 2 týdny.

2 Popis a rozbor problému

Tato úloha se věnuje řešení praktického problému konvexní obálky pro náhodné množiny bodů, pravidelné množiny bodů. Jako implementaci si lze zjednodušeně představit vytvarování digitální mapy.

3 Popisy algoritmů

V dané úloze jsou použity následující algoritmy, avšak existují i další možnosti, jak polohu bodu určit.

3.1 Jarvis Scan

Jedna se o dost pomalý algoritmus oprotí ostatním algoritmům zabývájicím se obdobním problémy. Algoritmus funguje na principu gip wrapping, česky balení dárku.
Nechť v kartezské soustavě souřadnic existuje množina bodů. Setřiděním souřadnic dle osy $\boldsymbol{Y}$, je možné najít počáteční bod $\boldsymbol{q}$ (dále jen pivot), který je dan obrázek bodu s nejmenší hodnotou na ose $\boldsymbol{Y}$. Dále předpokladáme, že v množině uvedených bodů neexistují 3 kolineární body. Pak je možné aplikovat níže uvedený algoritmus:

1. Nalezení pivota $q = \min(y_i)$,
2. Vklad q do množiny H.
3. p_j = q, $p_i = p_{j-1}$.
4. Načítání bodu p_{j-1}
5. $p_i \neq$ q:
6. Cyklus pro body p_{j-1}, p_j:
7. Dokud p_i je takové, že $\theta = \min(\theta_i)$;
8. Vklad do množiny H.

3.2 Quick Hull

Algoritmus typu QuikSort. Konvexní obálka tvořena z horní a dolní částí.
Horní část obsahuje body nad spojnicí bodů a dolní část je pod spojnicí q_1 a q_3.
Řešení pro každou konvexní obálku se provádí zvlašť a následně se sloučuje. Hledáme nejvzdalenější bod pro každou konvexní obálku ležící vpravo od této strany. Nově vzniklý bod se stává bodem obálky. Nově vzniklá strana se rozděluje na dvě nové strany, princip rozděluj a panuj.

3.3 Incremental Construcion

Inkrementální algoritmus je velmi rychlý algoritmus pro výpočet konvexní obalky. Princip se základa na postupným testováním bodů, respektive jejich předaváním do konvexní obalky, jejiž tvar je modefikovan.

3.4 Graham Scan

Grahamuv Scan je algoritmus určený pro výpočet konvexní obálky ve dvou rozměrném prostoru.
Algoritmus nejprve vyhledává pivot p , jehož souřadnice y je minimální. Z něj se proloži polární orientace na ostatní body. Body jsou následně seřazeny dle velikostí těchto orientaci, respektivě úhlů. Pivot a všechny body co leží vlevo, jsou postupně ukládany do konvexní obálky.

3.5 Striktně konvexní obálka

Účelem algoritmu je detikovat body na přímkach, respektivě, aby přimka jako taková, byla tvořena jen 2 body, tedy počátkem a koncem.
V algoritmu jsou postupně procházeny všechny trojíce bodů. Pokud se prokazalo, že 3 body leží na společní přímce, tak prostřední bod je detikovan.

4 Vstupní data

Aplikace má 2 mody. První mod znázorňuje výsledky výpočtu obrázem situace.Mezi vstupními daty patří jediná hodnota, a to koknretně uživatelem zadaná hodnota požadovaného počtu bodů. Generovát lze na základě randomného, respektve náhodného rozložení, nebo na základě gridu, respektive pravidelné síti bodů. V rámci třetí možností je třeba ještě uvést náhodného generování shluku bodů. Druhý mod, respektive Graph

mode vytváří graf, který znázorňuje generování v konkretních metodach v intervalu od 1000 do 1000000 a úkláda výsledky grafů do formatu .PDF.

5 Výstupní data

Výstup je vizualizací řešeného problému v grafickém okně. Taky v grafickém okně je napsána rychlost výpočtu algoritmu.
Měření času probíhá v programu vždy při spuštění konkrétního algoritmu (a pouze něj - neměří se rychlost generování bodů ani jejich vykreslení. Pouze samotný výpočet obálky). Před spuštěním algoritmu a po jeho ukončení se odečtou časové hodnoty pomocí `std::clock_t`, z jejichž rozdílu se poté zjistí tzv. *processor time*, jehož následným vydělením `CLOCKS_PER_SEC` získáme čas v sekundách.
Praktickým výstupem je graf znázorňující rychlot výpočtu algoritmu v podobě vykreslování grafu. Uživátel si sam zvolí umistění. Víc další kapitoly

6 Ukázky aplikace

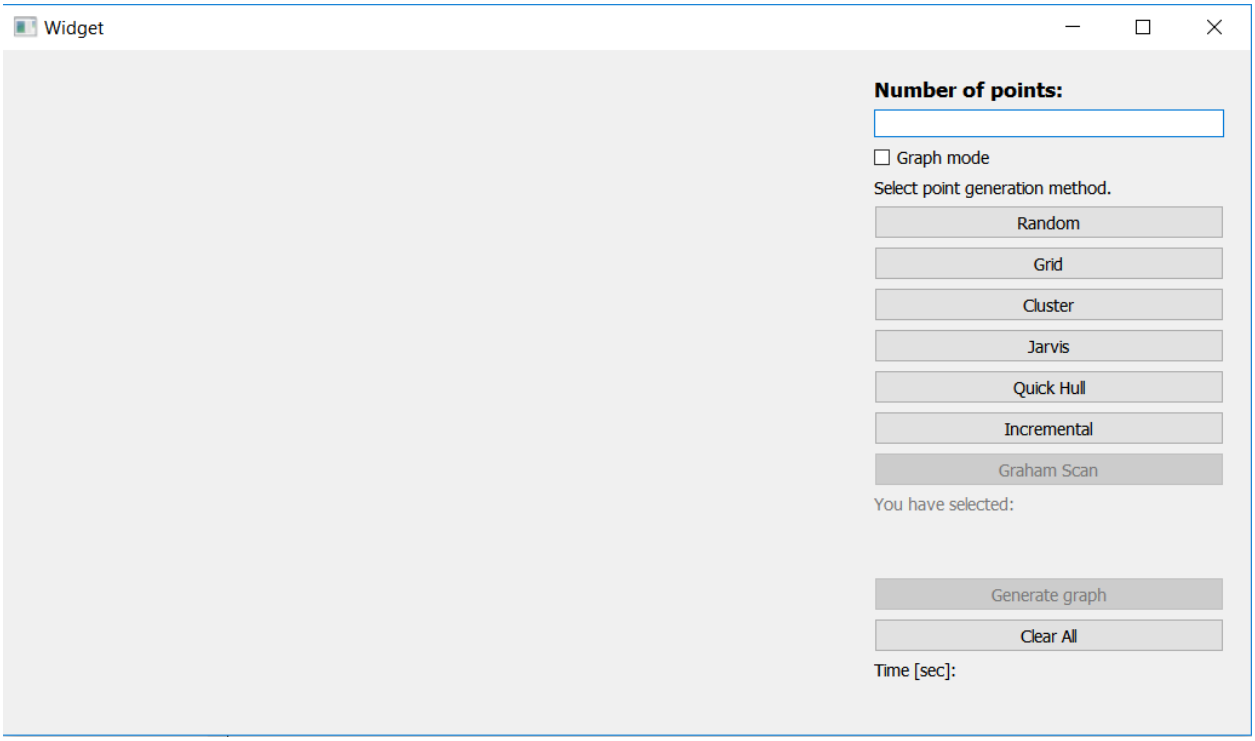


Figure 1: Aplikace po spuštění

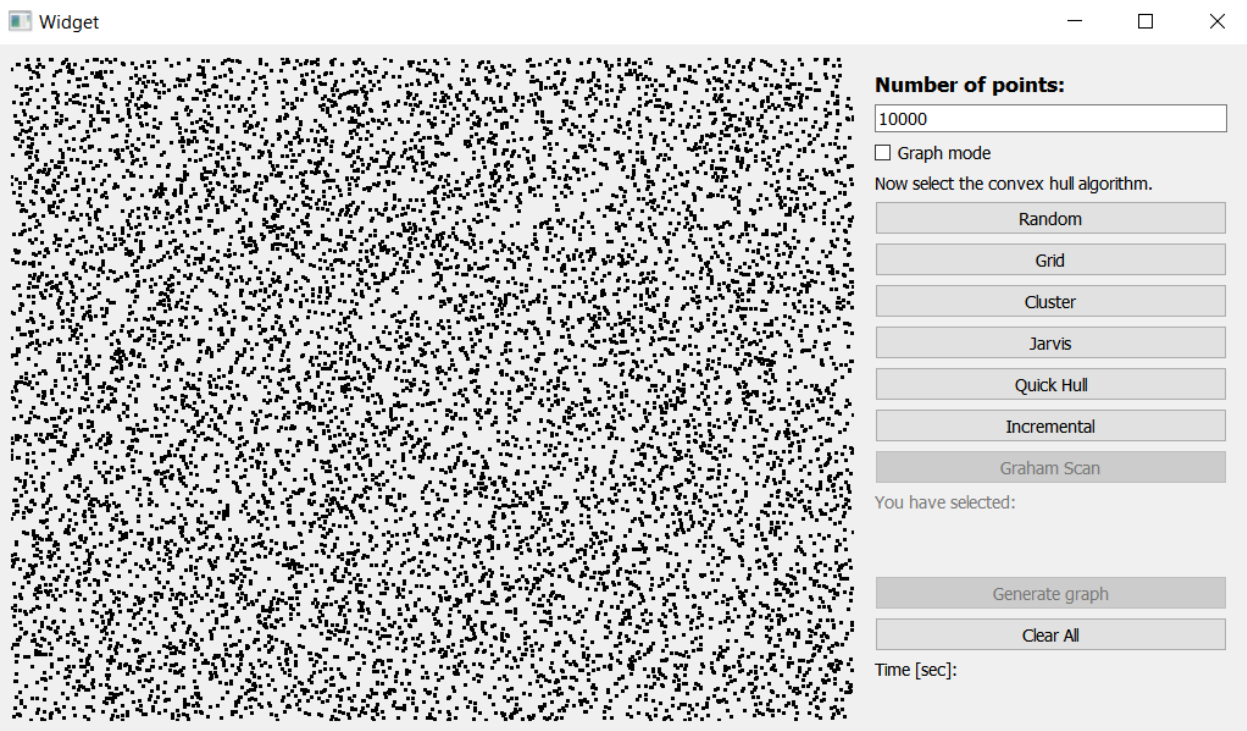


Figure 2: Aplikace po spuštění RANDOM

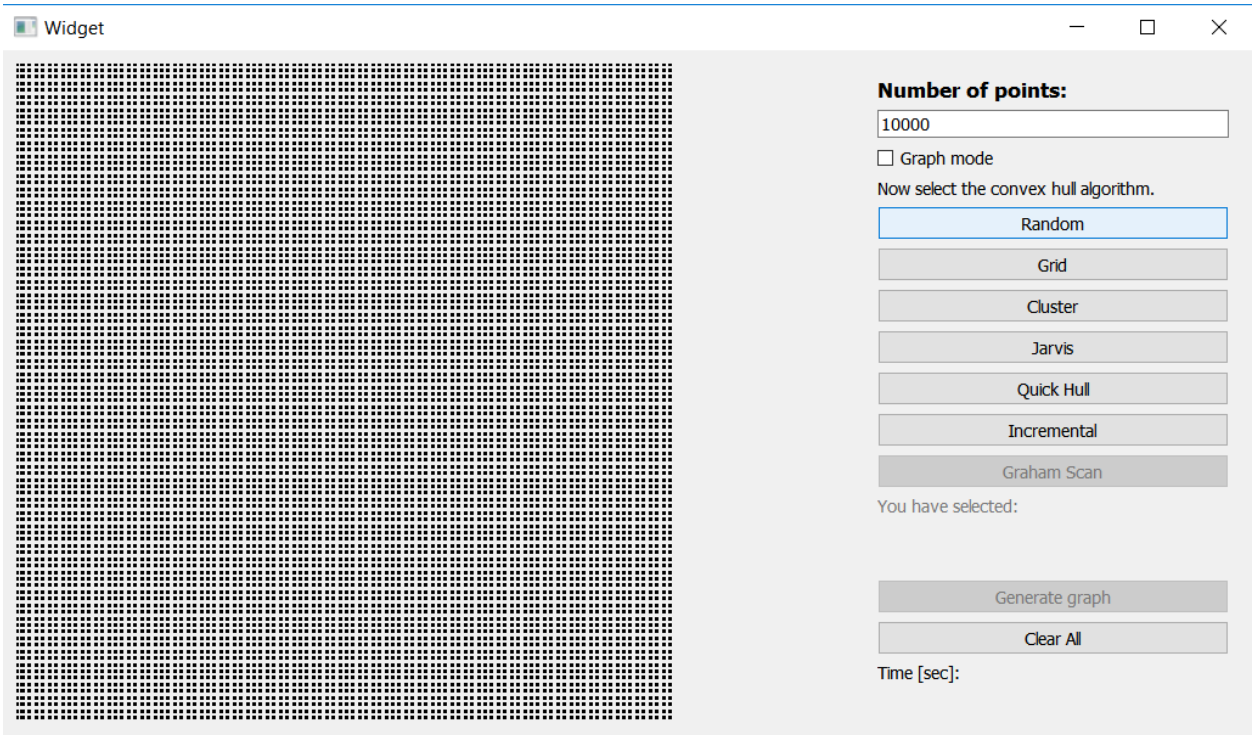


Figure 3: Aplikace po spuštění GRID

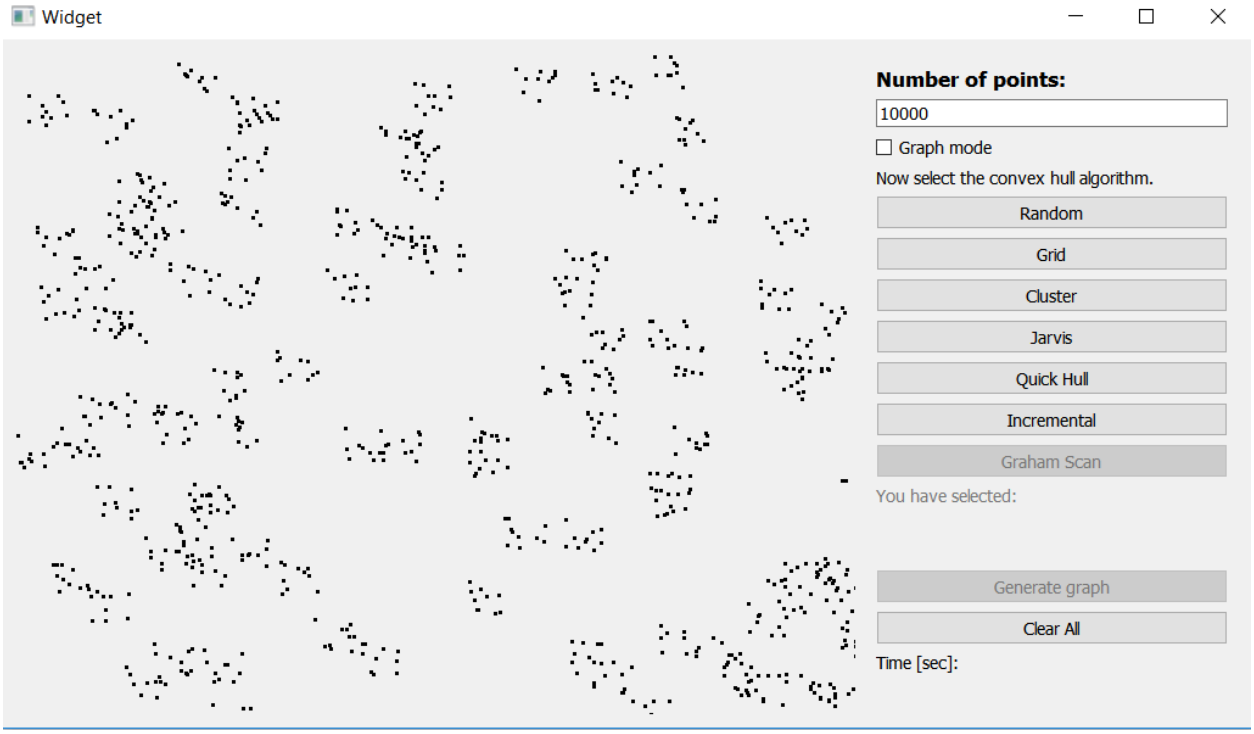
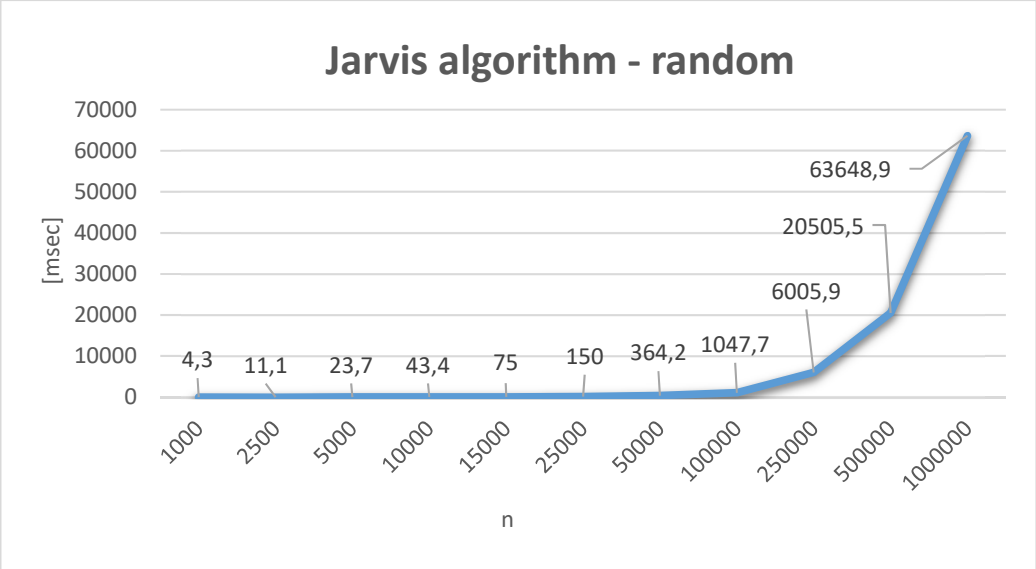


Figure 4: Aplikace po spuštění Cluster

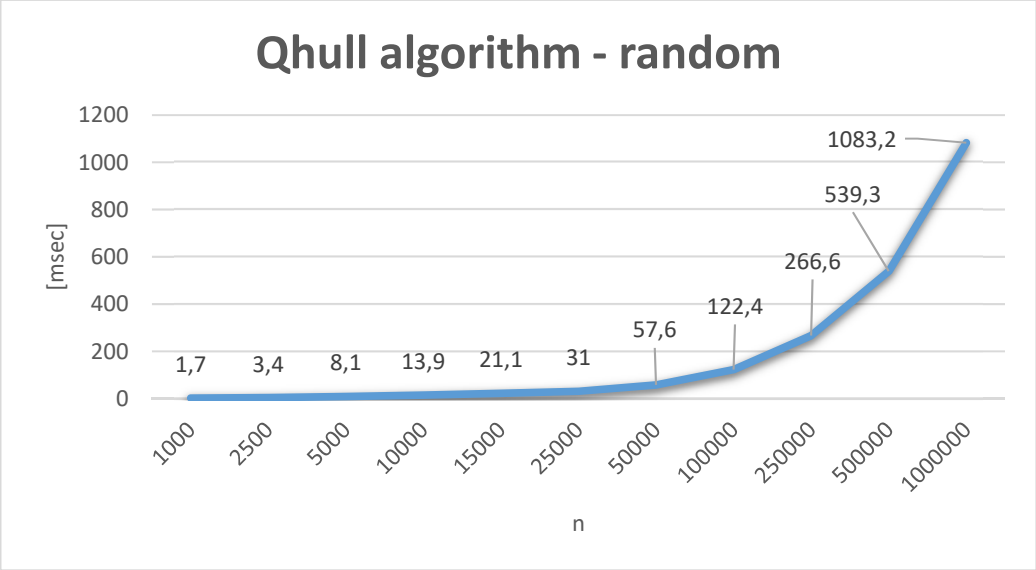
6.1 RANDOM

Jarvis Scan



pocet bodu	1000	2500	5000	10000	15000	25000	50000	100000	250000	500000	1000000
1	5	10	21	46	82	155	316	1107	5767	20168	68634
2	4	12	24	44	77	139	401	1004	6425	21544	71104
3	4	11	19	42	75	145	368	987	6225	19877	70819
4	4	10	26	39	74	147	354	1145	5785	20574	70450
5	4	12	28	45	76	158	325	1025	5598	20147	75576
6	5	11	26	44	69	139	398	1055	5586	20698	7854
7	4	13	26	43	75	142	364	988	6014	19899	64997
8	5	10	24	45	72	164	375	1068	6325	20147	69757
9	4	11	21	42	76	157	416	1024	6222	21004	65754
10	4	11	22	44	74	154	325	1074	6112	20997	71544
	4,3	11,1	23,7	43,4	75	150	364,2	1047,7	6005,9	20505,5	63648,9

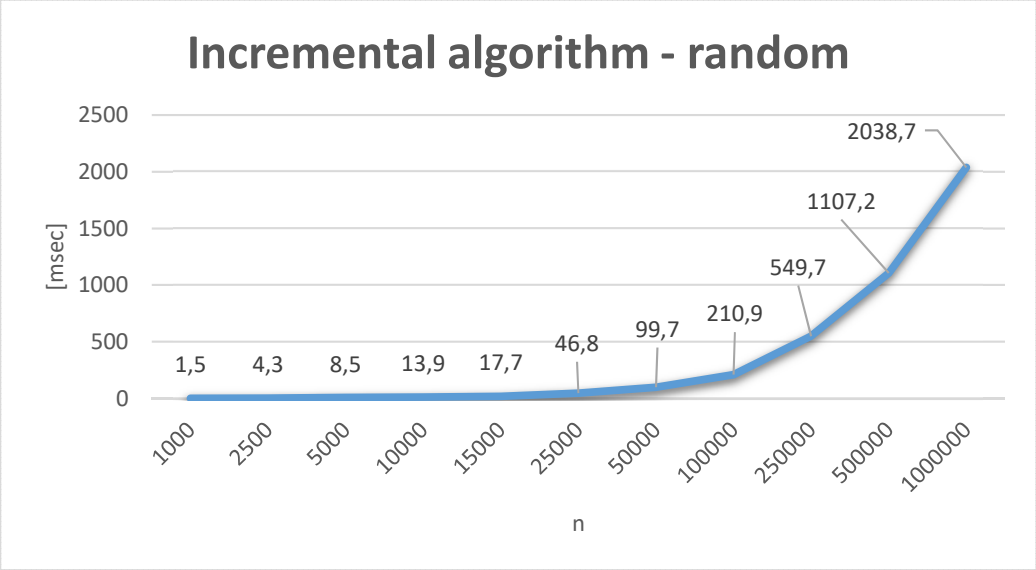
Quick Hull



pocet bodu	1000	2500	5000	10000	15000	25000	50000	100000	250000	500000	1000000
1	1	3	7	12	20	29	55	122	268	538	1018
2	2	3	8	15	19	31	53	115	254	555	1063
3	2	4	8	14	19	35	64	129	278	527	1120
4	2	4	8	15	21	28	61	124	271	511	1039
5	1	3	9	12	24	34	57	117	265	544	1079
6	2	3	9	13	23	32	54	128	259	523	1077
7	2	4	7	14	25	29	61	124	264	570	1089
8	2	3	9	14	21	33	59	131	269	529	1097
9	2	3	8	15	19	31	54	119	267	535	1151
10	1	4	8	15	20	28	58	115	271	561	1099
	1,7	3,4	8,1	13,9	21,1	31	57,6	122,4	266,6	539,3	1083,2

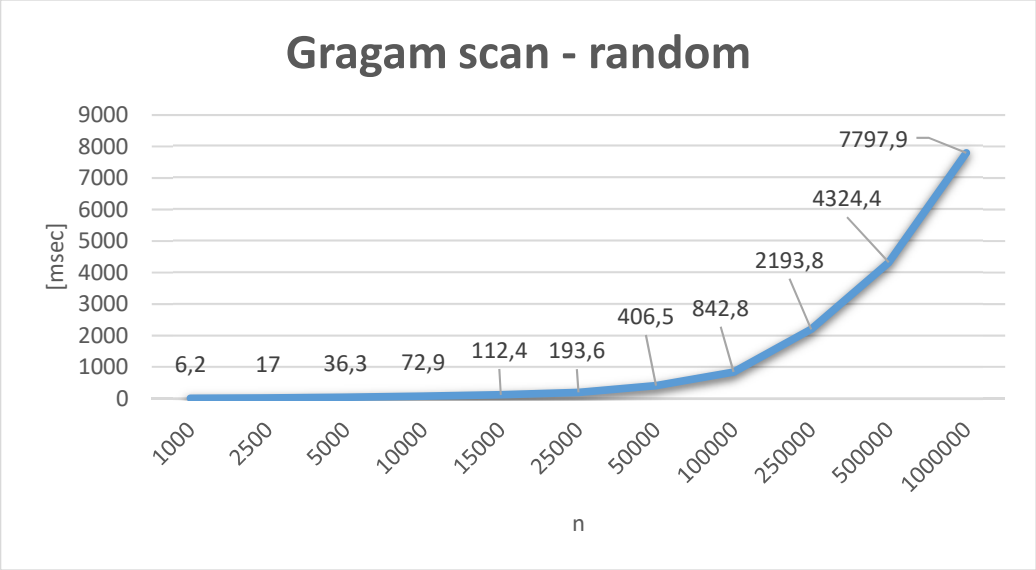
Incremental construction

.



pocet bodu	1000	2500	5000	10000	15000	25000	50000	100000	250000	500000	1000000
1	1	4	8	12	17	46	99	202	549	1081	1994
2	1	4	8	15	17	48	99	219	555	1097	2058
3	2	4	8	14	19	47	101	211	541	1131	2092
4	1	4	9	15	18	47	100	214	547	1122	2038
5	1	5	9	12	18	46	101	209	557	1095	2041
6	1	5	8	13	17	45	100	205	552	1097	1999
7	2	4	9	14	17	49	100	215	544	1121	2047
8	2	5	8	14	18	46	99	217	557	1125	2022
9	2	4	9	15	18	48	100	208	551	1108	2101
10	2	4	9	15	18	46	98	209	544	1095	1995
	1,5	4,3	8,5	13,9	17,7	46,8	99,7	210,9	549,7	1107,2	2038,7

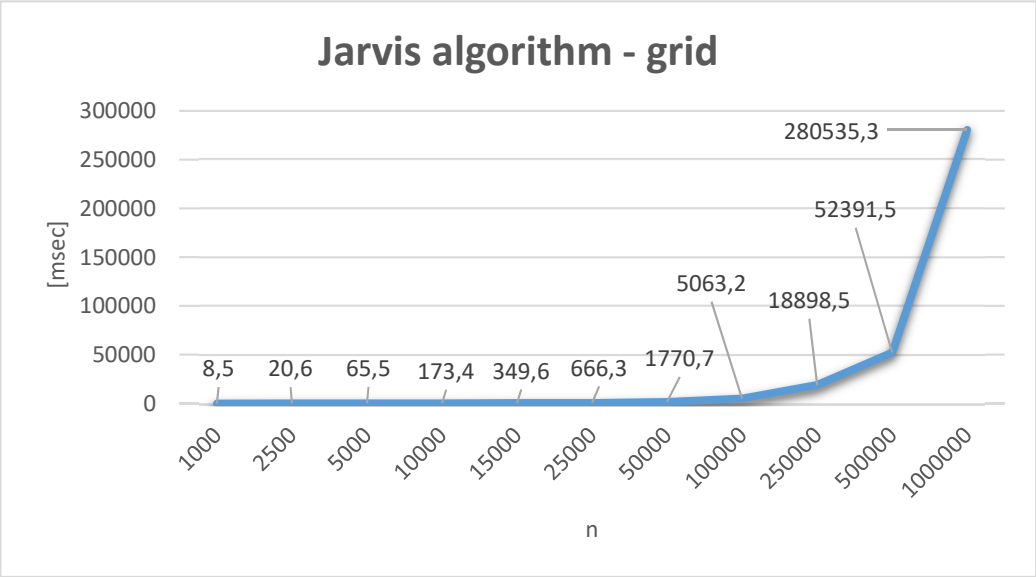
Graham Scan



pocet bodu	1000	2500	5000	10000	15000	25000	50000	100000	250000	500000	1000000
1	6	17	36	73	113	198	411	856	2204	4383	7874
2	6	17	35	72	112	194	400	852	2211	4198	7728
3	6	17	36	73	111	191	400	847	2187	4425	7874
4	6	17	36	74	113	190	395	820	2246	4328	7683
5	6	17	36	73	113	200	412	830	2231	4317	7885
6	7	17	36	72	112	188	415	834	2213	4296	7856
7	6	17	38	73	112	193	417	838	2165	4305	7684
8	7	17	36	73	113	195	405	848	2188	4300	7728
9	6	17	38	73	113	189	399	842	2134	4395	7832
10	6	17	36	73	112	198	411	861	2159	4297	7835
	6,2	17	36,3	72,9	112,4	193,6	406,5	842,8	2193,8	4324,4	7797,9

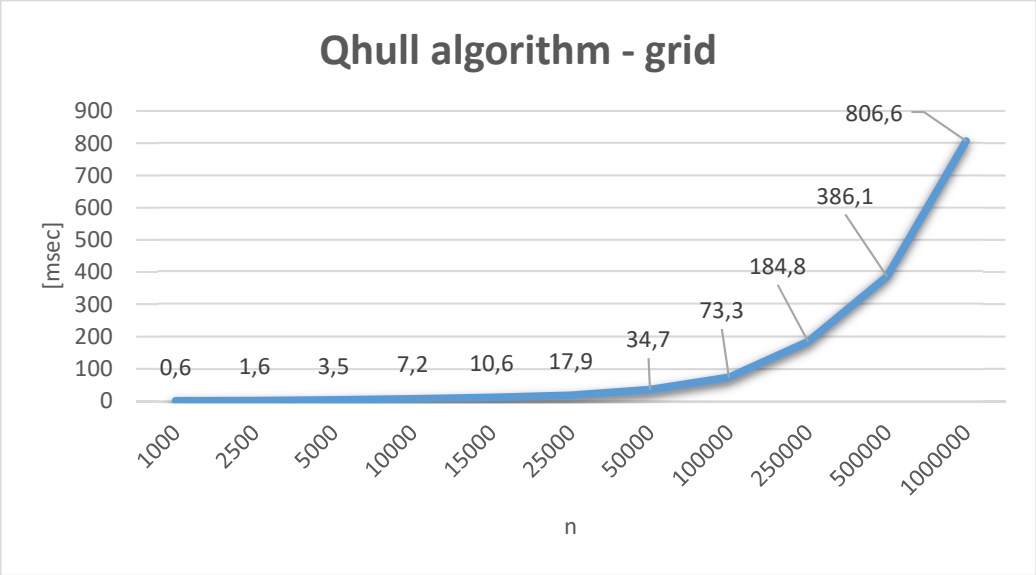
6.2 GRID

Jarvis Scan



pocet bodu	1000	2500	5000	10000	15000	25000	50000	100000	250000	500000	1000000
1	8	21	64	171	347	663	1763	5042	18839	52373	281982
2	8	21	67	175	351	669	1780	5043	18950	52315	279871
3	8	19	65	174	347	659	1778	5041	19078	52313	281449
4	9	22	65	175	349	658	1759	5092	18930	52521	280599
5	8	21	67	174	348	671	1777	5082	18871	52421	279565
6	9	21	66	171	347	675	1758	5075	18793	52401	281005
7	8	21	64	171	353	668	1767	5067	18879	52512	280339
8	9	19	65	178	352	671	1759	5062	18901	52390	280642
9	9	20	64	177	351	661	1789	5081	18954	52389	279562
10	9	21	68	168	351	668	1777	5047	18790	52280	280339
	8,5	20,6	65,5	173,4	349,6	666,3	1770,7	5063,2	18898,5	52391,5	280535,3

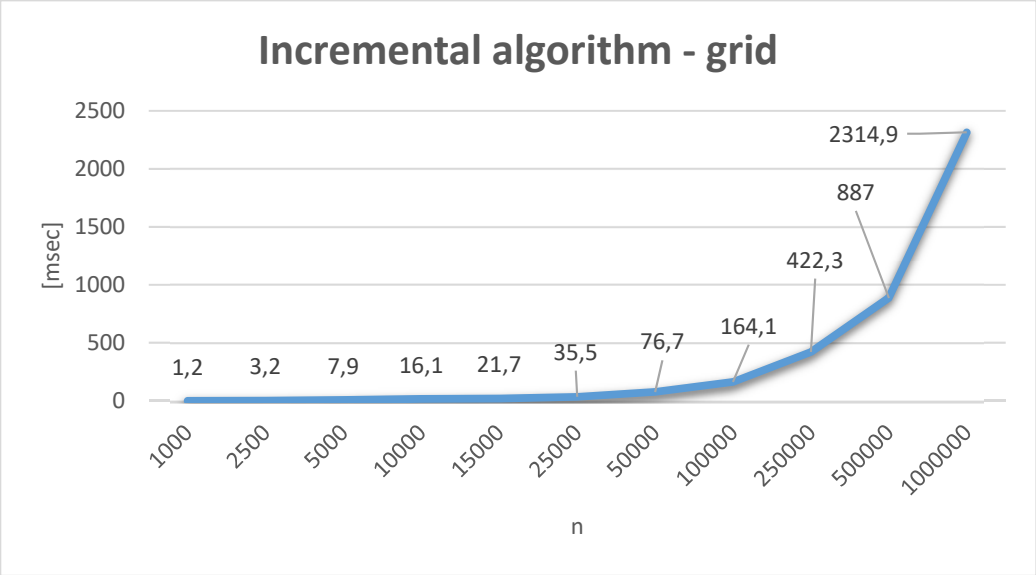
Quick Hull



pocet bodu	1000	2500	5000	10000	15000	25000	50000	100000	250000	500000	1000000
1	0	1	4	7	11	17	35	74	187	386	791
2	1	1	3	7	10	18	34	71	183	389	816
3	0	2	3	7	11	17	35	75	185	384	802
4	1	2	4	8	10	19	36	72	184	386	821
5	1	2	4	7	10	17	33	73	182	386	792
6	0	2	3	7	11	17	34	75	181	385	798
7	1	1	3	8	11	18	35	72	190	383	789
8	1	2	4	7	11	19	35	71	185	387	821
9	0	2	3	7	11	19	36	75	187	387	825
10	1	1	4	7	10	18	34	75	184	388	811
	0,6	1,6	3,5	7,2	10,6	17,9	34,7	73,3	184,8	386,1	806,6

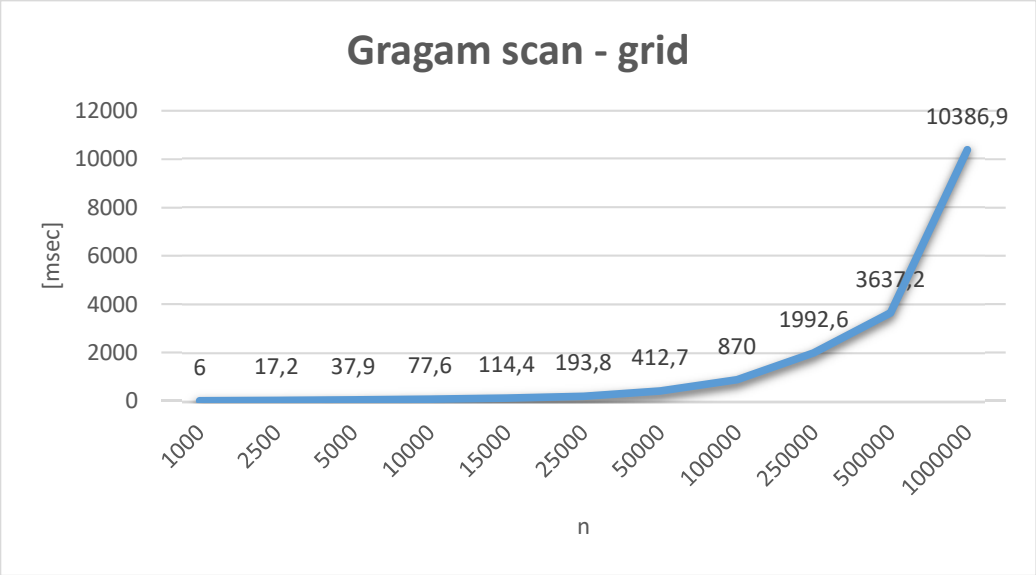
Incremental construction

.



pocet bodu	1000	2500	5000	10000	15000	25000	50000	100000	250000	500000	1000000
1	1	3	7	15	21	37	77	167	417	886	2333
2	2	3	8	15	22	35	78	161	424	897	2290
3	1	3	8	17	22	34	79	165	429	887	2317
4	1	4	8	18	22	36	74	165	415	878	2313
5	1	3	8	16	22	35	75	167	421	895	2331
6	1	3	9	18	22	35	76	168	424	880	2299
7	2	3	7	17	22	35	77	164	418	879	2324
8	1	3	8	14	21	36	77	159	425	892	2327
9	1	3	8	16	22	37	78	162	429	885	2315
10	1	4	8	15	21	35	76	163	421	891	2300
	1,2	3,2	7,9	16,1	21,7	35,5	76,7	164,1	422,3	887	2314,9

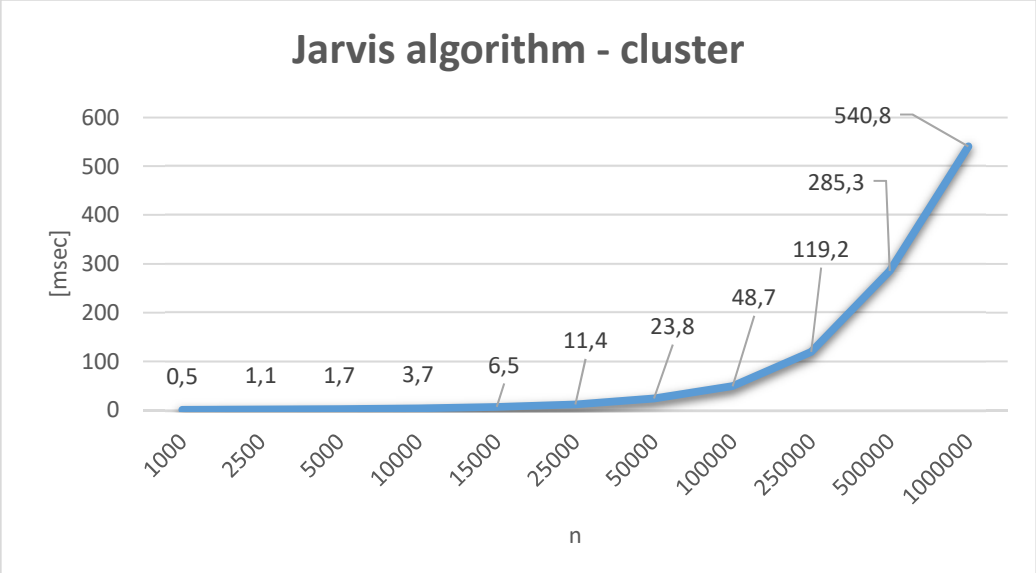
Graham Scan



pocet bodu	1000	2500	5000	10000	15000	25000	50000	100000	250000	500000	1000000
1	6	17	37	76	116	195	411	961	1956	3636	10237
2	6	17	38	79	113	193	415	838	1999	3615	10407
3	6	17	39	77	114	193	412	872	2004	3605	10386
4	6	18	38	78	114	193	412	861	1998	3654	10404
5	6	17	38	79	115	194	411	849	1988	3667	10539
6	6	17	37	79	116	194	415	855	2009	3663	10279
7	6	17	38	76	113	193	415	874	1954	3678	10348
8	6	17	38	77	113	194	411	881	1958	3589	10405
9	6	17	38	78	116	194	413	848	2046	3610	10485
10	6	18	38	77	114	195	412	861	2014	3655	10379
	6	17,2	37,9	77,6	114,4	193,8	412,7	870	1992,6	3637,2	10386,9

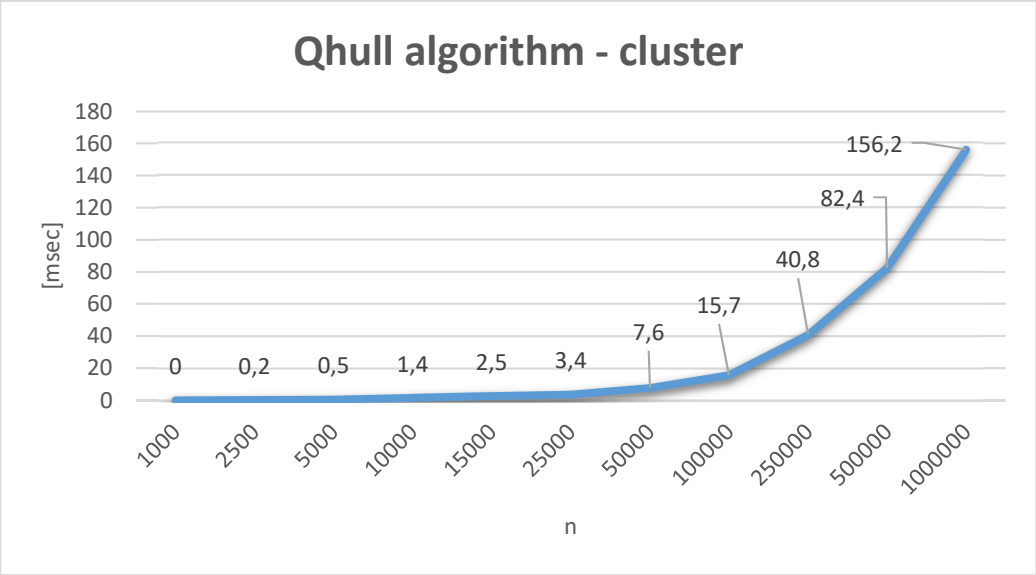
6.3 CLUSTER

Jarvis Scan



pocet bodu	1000	2500	5000	10000	15000	25000	50000	100000	250000	500000	1000000
1	0	1	2	4	6	12	27	48	122	254	545
2	0	0	1	4	7	13	20	45	131	248	500
3	0	2	2	4	7	11	24	51	125	301	598
4	1	1	2	4	6	10	21	50	119	268	601
5	1	1	2	3	6	11	23	49	120	298	594
6	0	2	2	4	7	12	24	50	115	315	584
7	1	1	2	4	6	11	25	48	108	275	521
8	0	1	1	3	7	10	25	47	129	325	497
9	1	1	1	3	6	13	24	50	101	301	521
10	1	1	2	4	7	11	25	49	122	268	447
	0,5	1,1	1,7	3,7	6,5	11,4	23,8	48,7	119,2	285,3	540,8

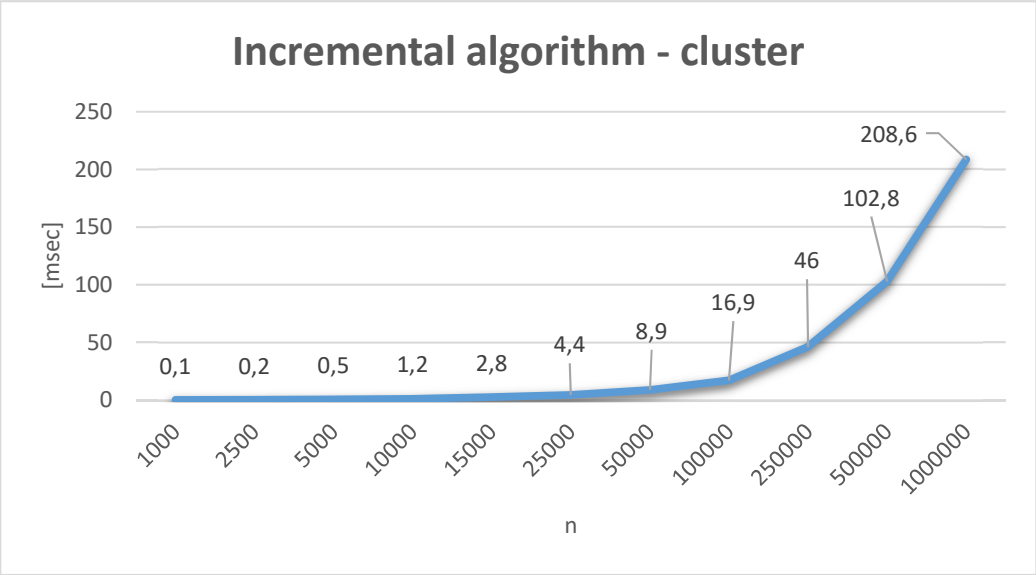
Quick Hull



pocet bodu	1000	2500	5000	10000	15000	25000	50000	100000	250000	500000	1000000
1	0	0	1	1	2	4	8	15	42	86	153
2	0	0	0	2	2	4	8	17	40	81	161
3	0	0	0	2	2	3	7	14	39	75	157
4	0	0	0	2	3	3	8	15	38	84	164
5	0	1	1	1	3	3	7	16	41	82	148
6	0	0	1	1	2	4	8	14	42	79	155
7	0	0	0	1	3	3	7	15	43	78	162
8	0	0	1	2	2	3	8	19	44	84	149
9	0	0	0	1	3	4	7	17	39	86	154
10	0	1	1	1	3	3	8	15	40	89	159
	0	0,2	0,5	1,4	2,5	3,4	7,6	15,7	40,8	82,4	156,2

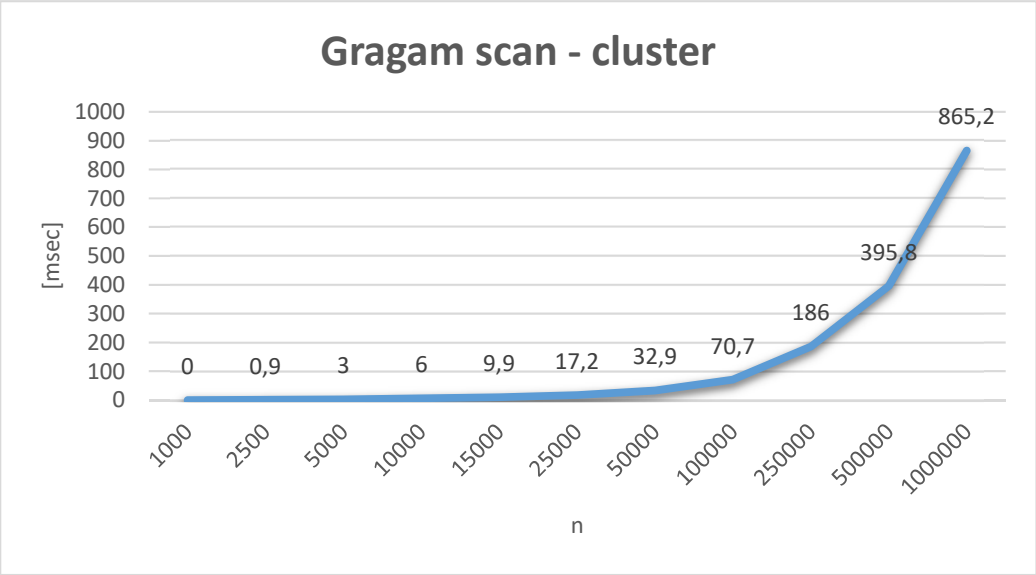
Incremental construction

.



pocet bodu	1000	2500	5000	10000	15000	25000	50000	100000	250000	500000	1000000
1	0	1	1	2	3	4	9	18	45	100	201
2	0	0	0	1	3	4	8	16	46	101	201
3	0	0	1	1	3	4	9	16	45	98	203
4	0	0	1	1	2	5	8	17	45	99	216
5	0	0	0	1	3	4	9	16	47	102	209
6	0	0	0	2	2	4	9	19	48	100	207
7	0	0	0	1	3	5	9	16	45	108	204
8	0	0	1	1	3	5	9	17	48	97	218
9	0	1	1	1	3	4	10	18	45	115	216
10	1	0	0	1	3	5	9	16	46	108	211
	0,1	0,2	0,5	1,2	2,8	4,4	8,9	16,9	46	102,8	208,6

Graham Scan



pocet bodu	1000	2500	5000	10000	15000	25000	50000	100000	250000	500000	1000000
1	0	1	3	6	10	16	33	70	183	404	831
2	0	1	3	6	10	17	33	73	188	394	845
3	0	1	3	6	10	17	34	68	190	389	897
4	0	1	3	6	10	18	32	69	184	395	903
5	0	1	3	6	10	19	33	71	183	401	856
6	0	1	3	6	9	16	34	70	188	395	874
7	0	1	3	6	11	16	31	72	182	396	868
8	0	1	3	6	9	18	32	70	191	391	851
9	0	1	3	6	10	17	33	71	185	388	835
10	0	0	3	6	10	18	34	73	186	405	892
	0	0,9	3	6	9,9	17,2	32,9	70,7	186	395,8	865,2

7 Závěr

Autoři splnili většinu bodů zadání a vznikl prográm, který generuje soubor bodů, jejich rozmístění a počita rychlost výpočtu algortimu. V předchozí kapitole jsou zobrazeny grafy, které znázorňují pruběhy výpočtu algoritmů. Test byl proveden několikrát. Jak je vidět z grafů, nejvíce nestabiliní je Jarvis Scan. Dochází k velkému kolisání v počátečných aj dokonce u středních hodnot. O něco lépe je na tom Quick Hull, a však aj u něho dochází ke znáčným výchylům. Obecně dalo by se řici, že Incremental construction je nejvíc stabilní algoritmus ze všech výše uvedených. Každopádně k výchylům na intervalu hodnot kolem několika tisíc dochází pravidelně a u všech algoritmů. S rostoucím počtem n klesa zároveň výchylka mezi jednotlivými testy. Zároveň s rostoucím počtem n klesa rozdíl mezi jednotilivými metodami. Z výsledku lze usoudit, že pro generování menšího objemu dat je lepší využit Incremental construction, případně Quick Hall, naopak pro velký objem dat co převýšuje velikosti desítek tisíc je možné využivát jaký koliv algoritmus. Graham Scan se prokazuje jakož to nevhodný a zpomalený algoritmus. Z bonusových úloh bylo naprogramováno:
Graham Scan
Konstrukce striktně konvexních obálek pro všechny algoritmy
Ošetření singulárního připadu u Jarvis Scan

7.1 Náměty na vylepšení

Aplikace, ač funkční a splňující daný účel, má spoustu nedostatků, které by bylo dobré v budoucnu odstranit. Autoři zde uvádí pár těch nejzjevnějších.
Souřadnicové osy:

Vykreslovací okno má v Qt, stejně jako ve většině podobných nástrojů, počátek souřadnic v levém horním rohu, kladnou osu x vpravo a kladnou osu y směrem dolů. Tento model se však neshoduje ani s geodetickými souřadnicemi používanými na našem území (kladná y doleva, kladná x dolů), ani s klasickým označením os (kladná x doprava, kladná y nahoru). Proto se body v současné verzi zobrazují jinak, než by možná uživatel očekával. Vhodným řešením by byla transformace.