

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
4	Vstupní data	7
5	Výstupní data	8
6	Ukázky aplikace	9
6.1	RANDOM	12
6.2	GRID	15
6.3	CLUSTER	18
7	Závěr	21
7.1	Náměty na vylepšení	21
	References	21

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, rozpracována konstrukce konvexní obálky metodou Graham Scan.

Úloha č. 2: Konvexní obálky a jejich konstrukce

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

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 $n \in \langle 1000, 1000000 \rangle$ 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 zabývajícím se obdobným problémem. Algoritmus funguje na principu gip wrapping, česky balení dárku.

Nechť v kartezské soustavě souřadnic existuje množina bodů. Setříděním souřadnic dle osy \mathbf{Y} , je možné

najít počáteční bod \mathbf{q} (dále jen pivot), který je dan obrázek bodu s nejmenší hodnotou na ose \mathbf{Y} . Dále předpokládáme, že v množině uvedených bodů neexistují 3 kolineární body. Pak je možné aplikovat níže uvedený algoritmus:

1. Nalezení pivotu $q = \min(y_i)$,
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 spojnici q_1 a q_3 .

Řešení pro každou konvexní obálku se provádí zvlášť 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 základá na postupným testováním bodů, respektive jejich předáváním do konvexní obalky, jejíž tvar je modifikovan.

4 Vstupní data

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

mode vytváří graf, který znázorňuje generování v konkrétních metodách v intervalu od 1000 do 1000000 a ukládá výsledky grafů do formátu .PDF.

5 Výstupní data

Výstup je vizualizací řešeného problému v grafickém okně. Taky v grafickém okně je slovně napsána rychlost výpočtu algoritmu.

Praktickým výstupem je graf znázorňující rychlost výpočtu algoritmu v podobě vykreslování grafu. Uživatel si sam zvolí umístění. Víc další kapitoly

6 Ukázky aplikace

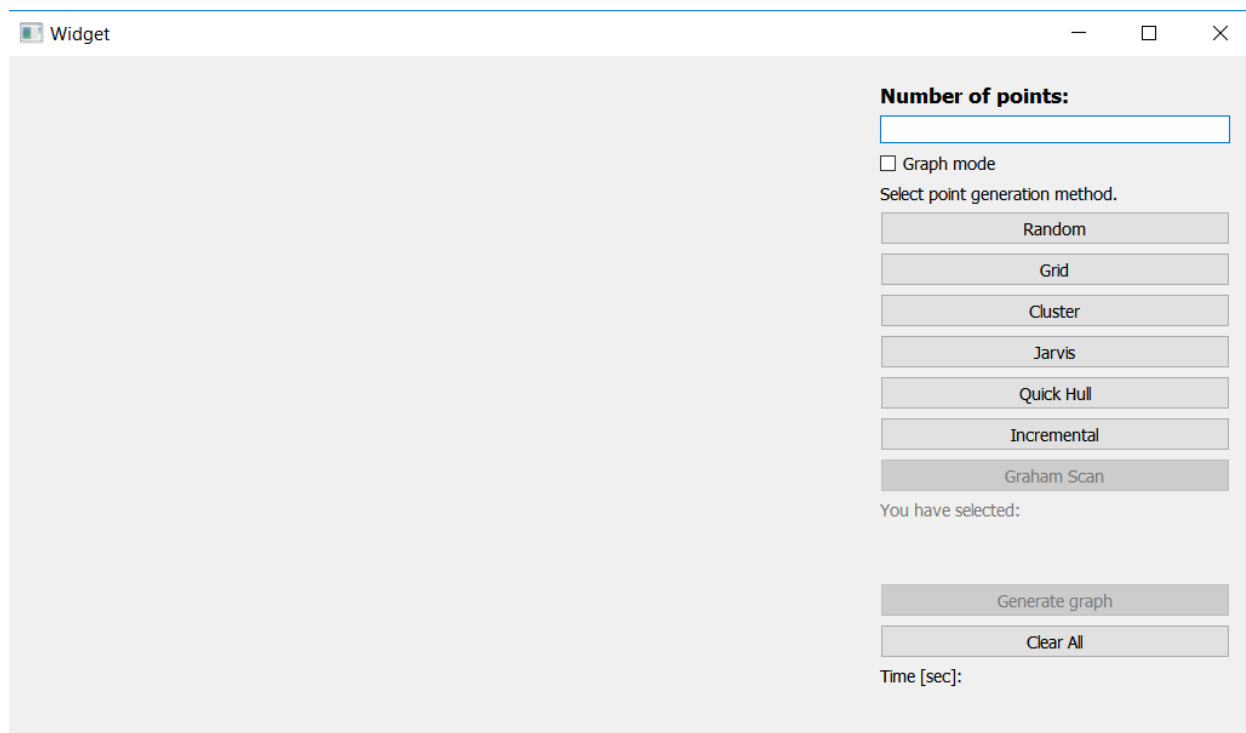


Figure 1: Aplikace po spuštění

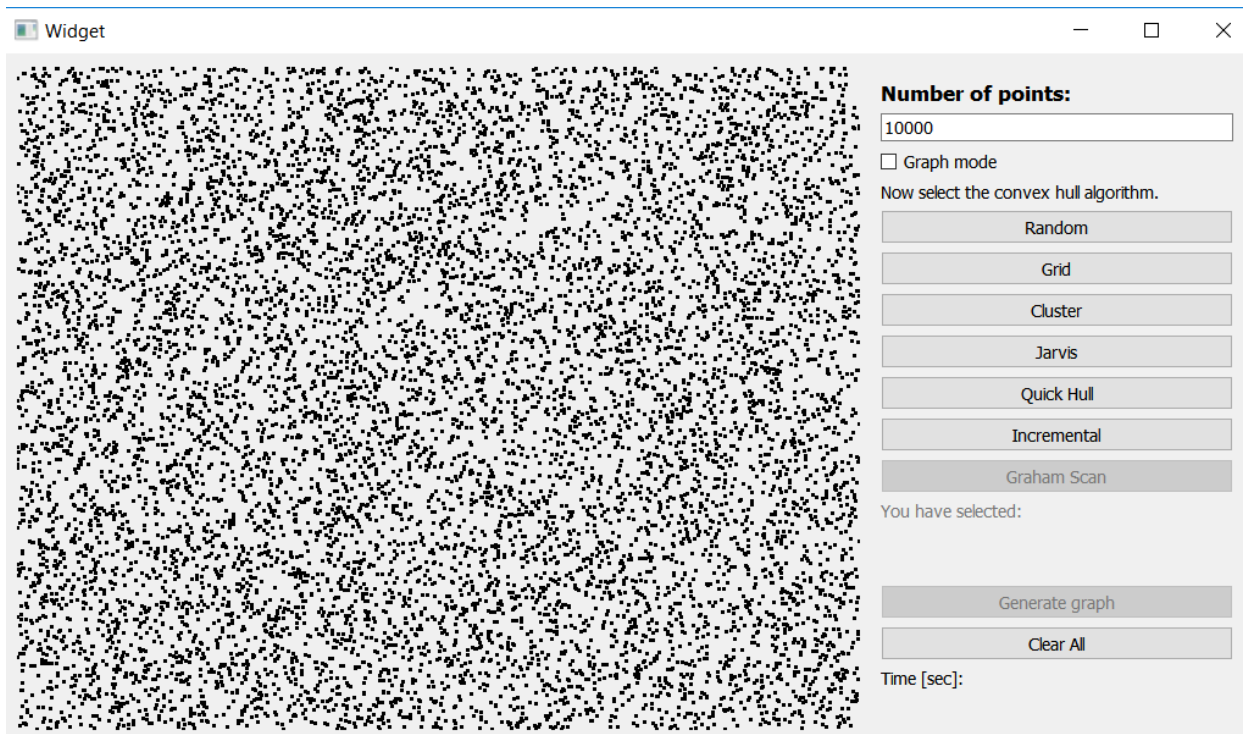


Figure 2: Aplikace po spuštění RANDOM

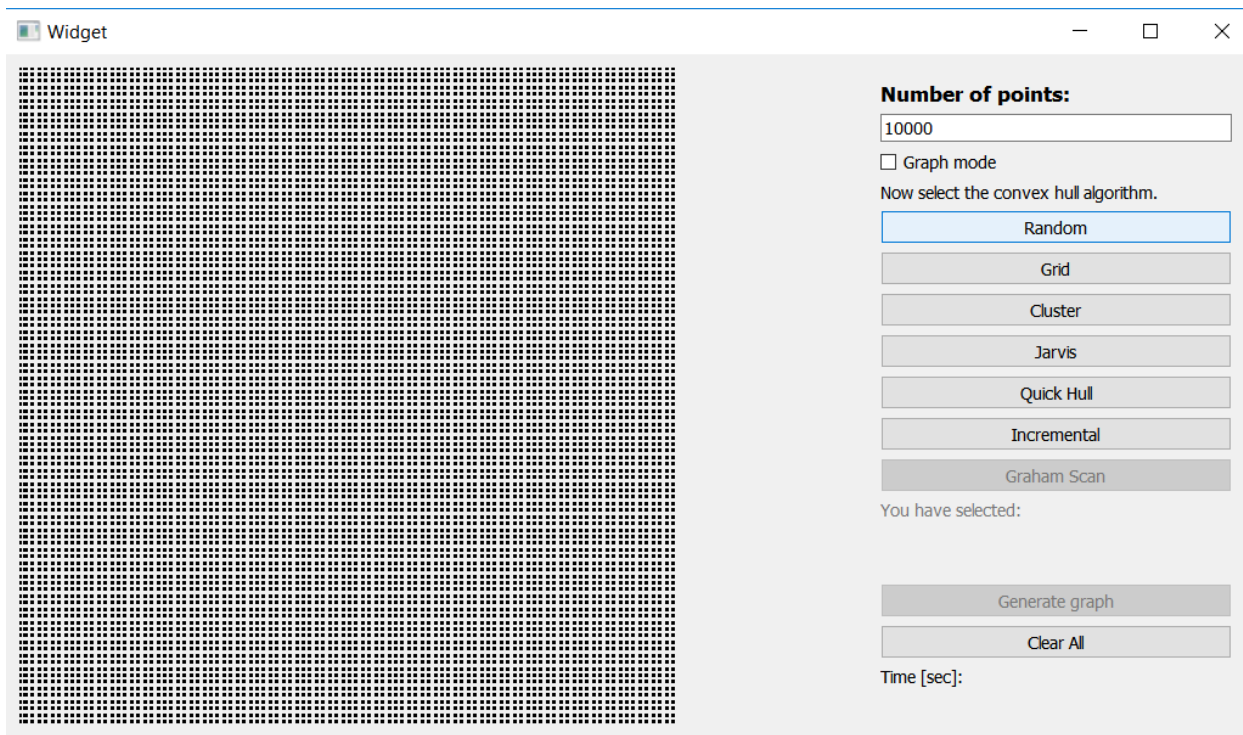


Figure 3: Aplikace po spuštění GRID

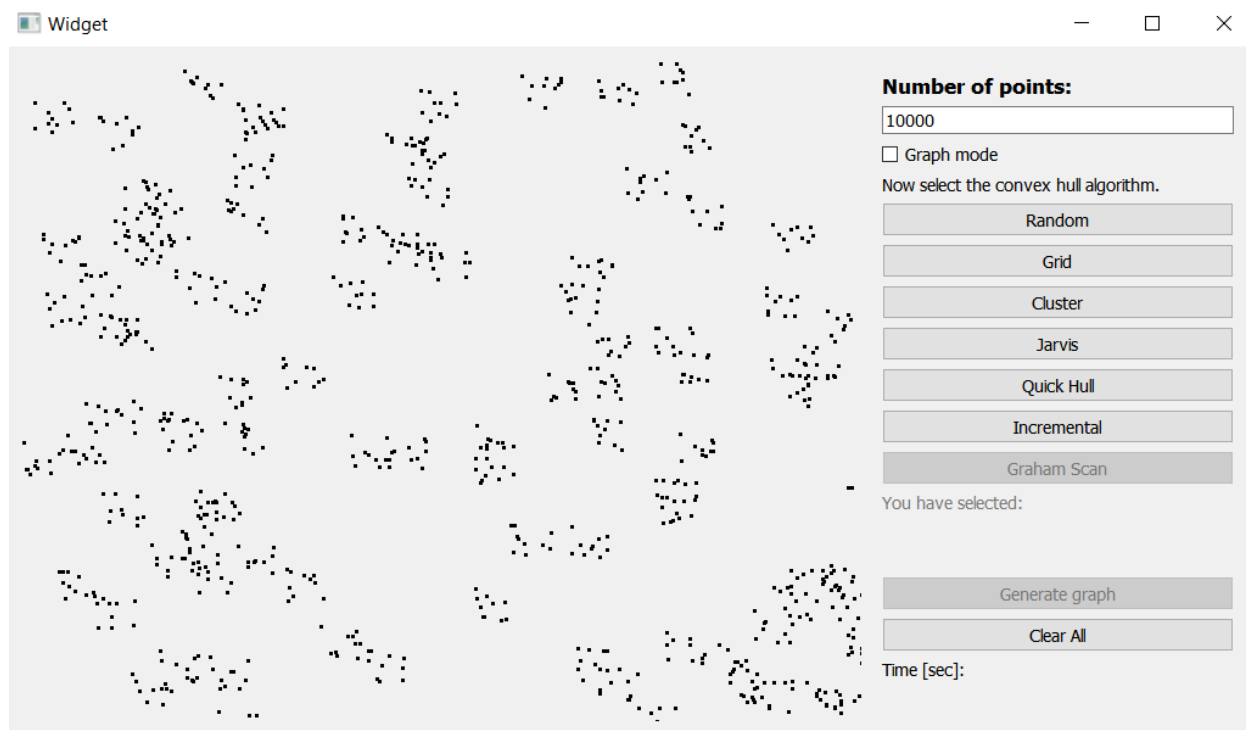


Figure 4: Aplikace po spuštění Cluster

6.1 RANDOM

Jarvis Scan

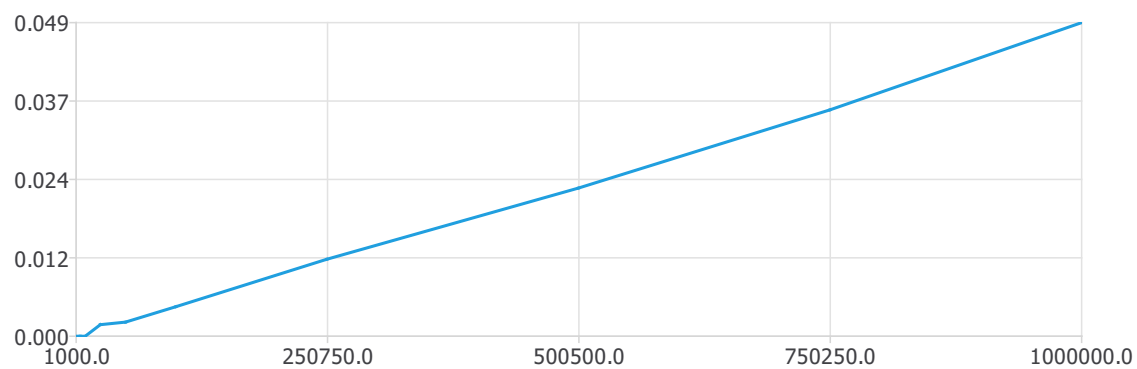


Figure 5: generování jednou

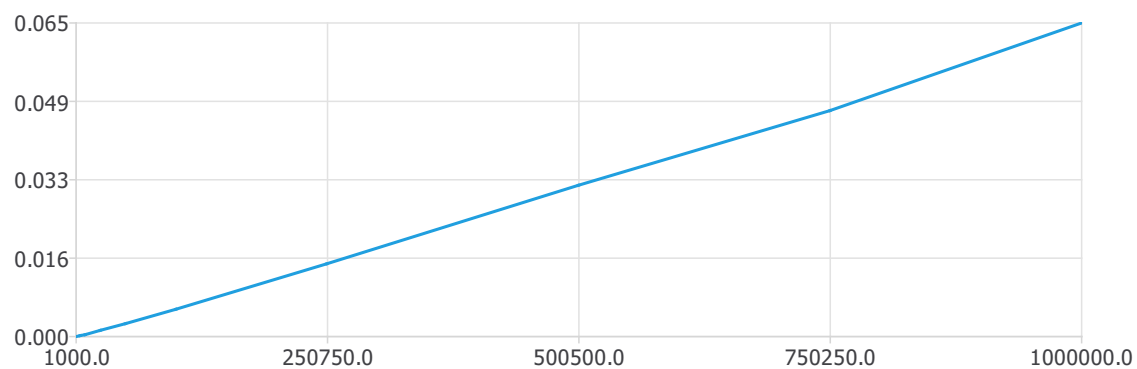


Figure 6: generování 10x

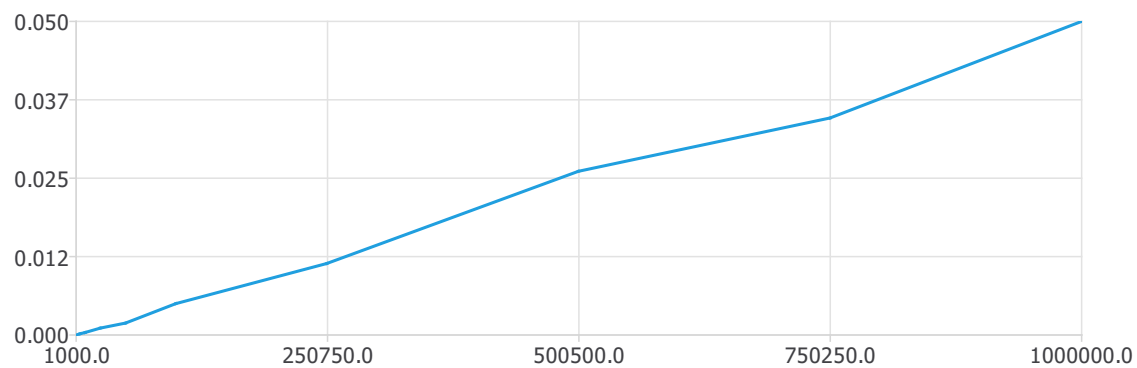
Quick Hull

Figure 7: generování

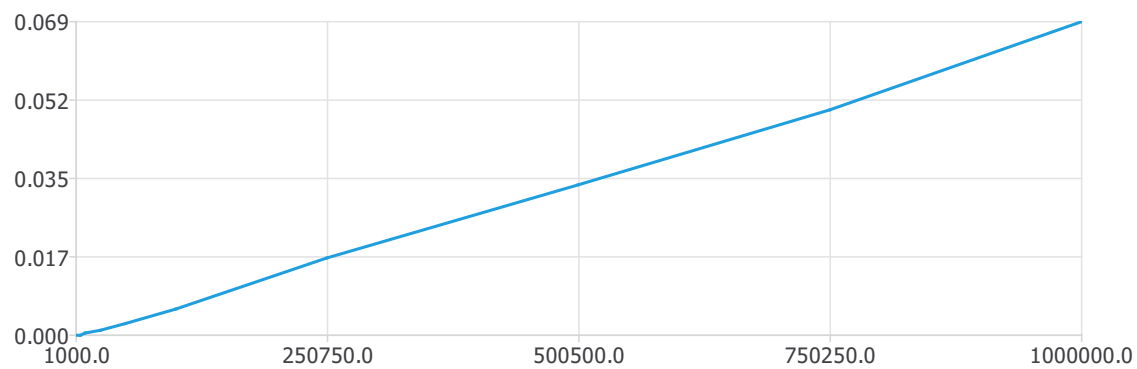


Figure 8: generování 10x

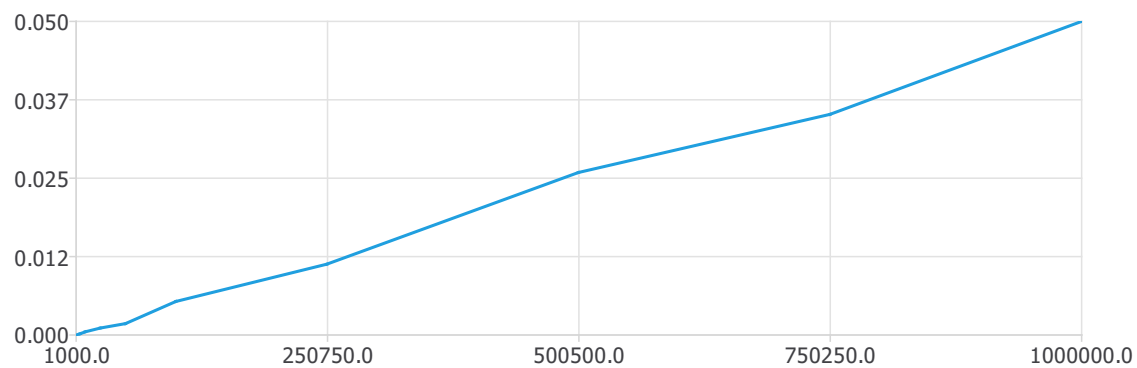
Incremental construction

Figure 9: generování

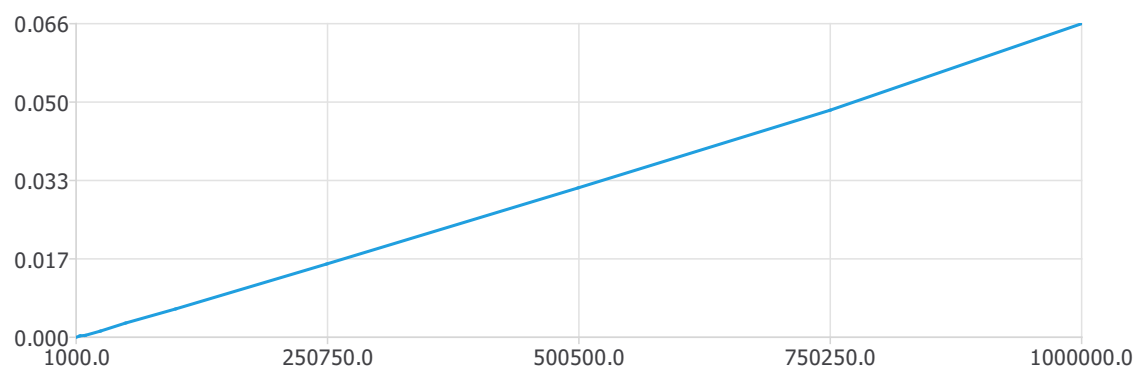


Figure 10: generování 10x

6.2 GRID

Jarvis Scan

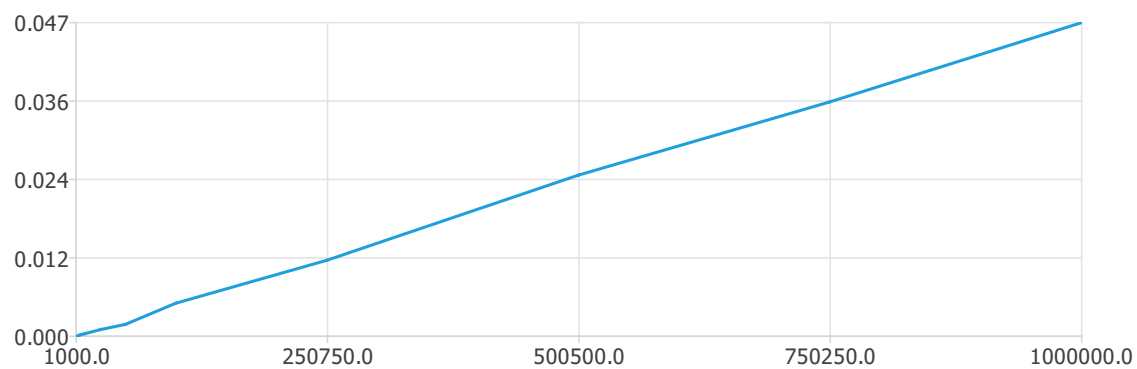


Figure 11: generování

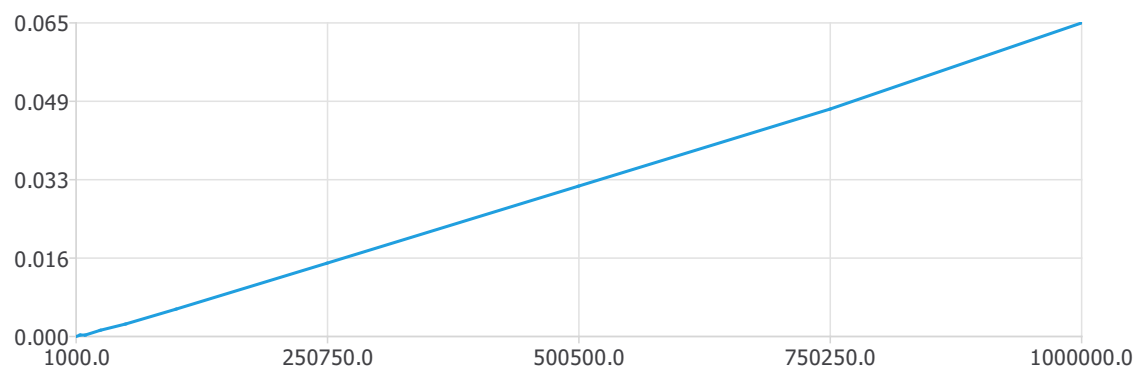


Figure 12: generování 10x

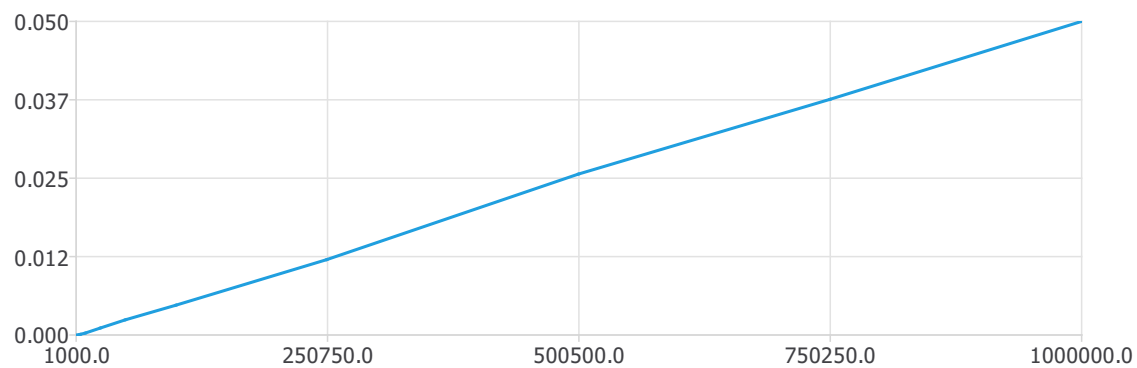
Quick Hull

Figure 13: generování

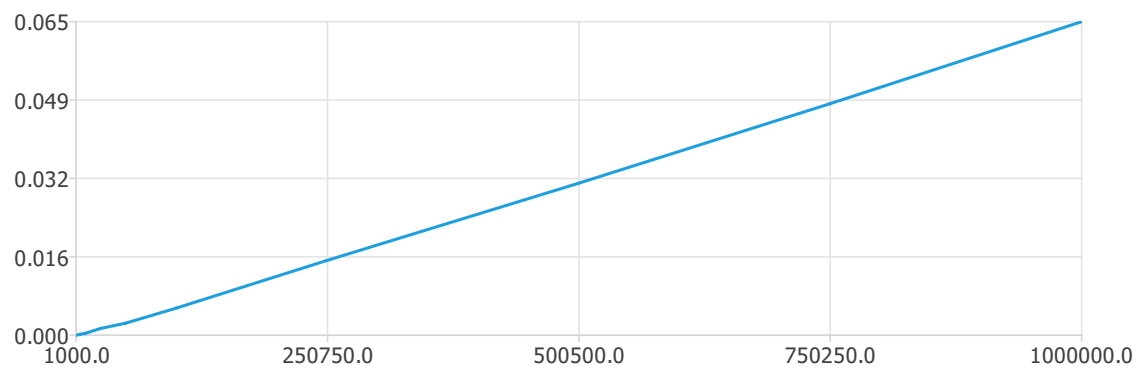


Figure 14: generování 10x

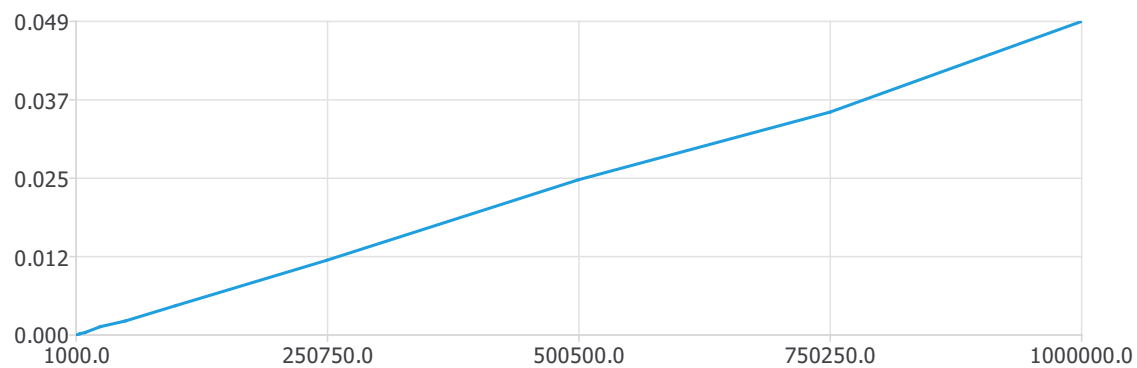
Incremental construction

Figure 15: generování

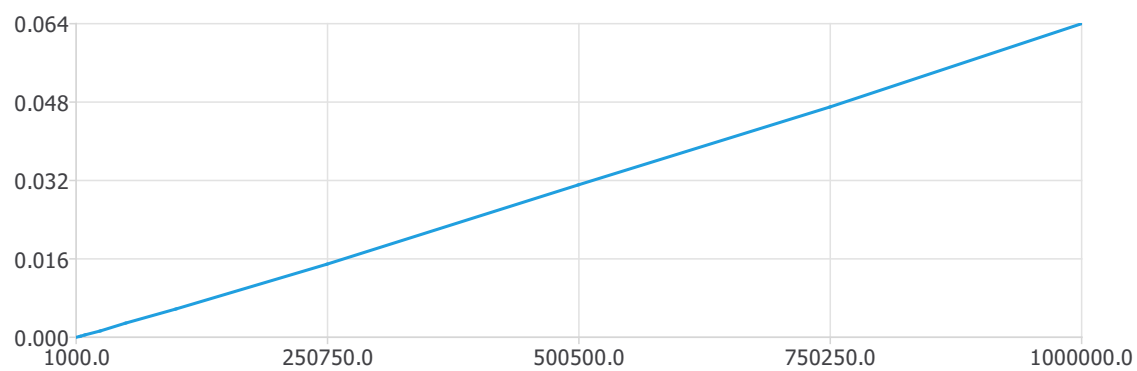


Figure 16: generování 10x

6.3 CLUSTER

Jarvis Scan

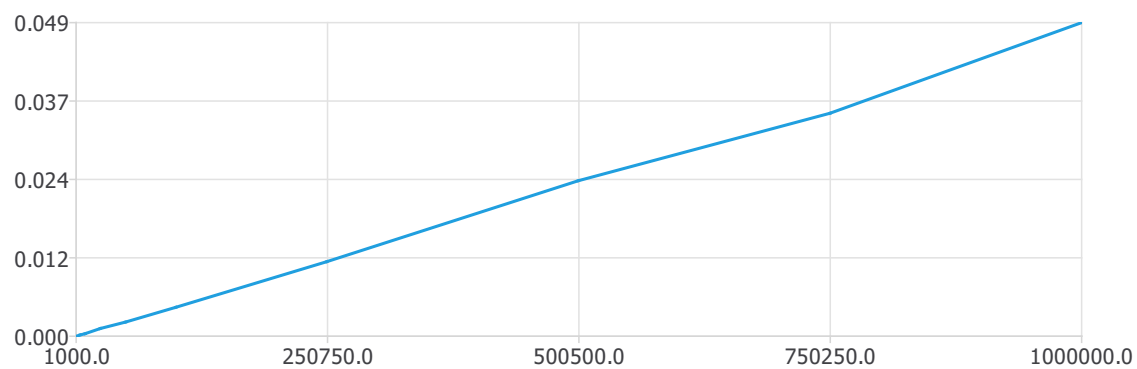


Figure 17: generování

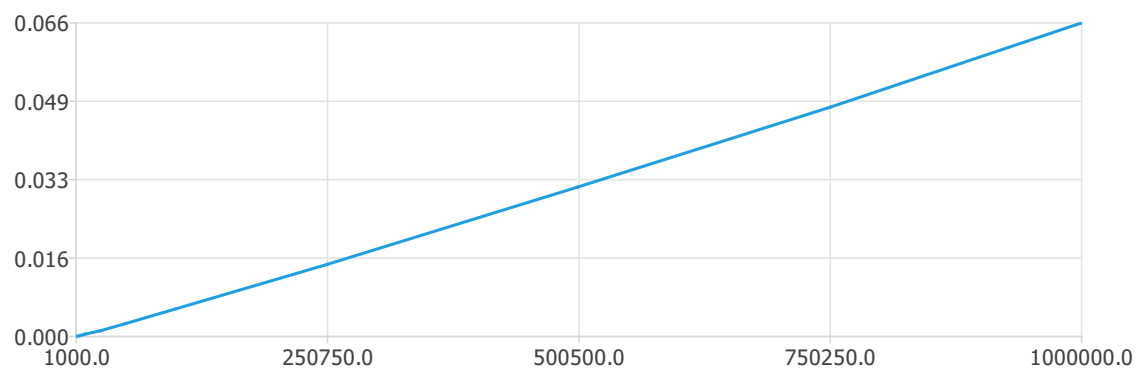


Figure 18: generování 10x

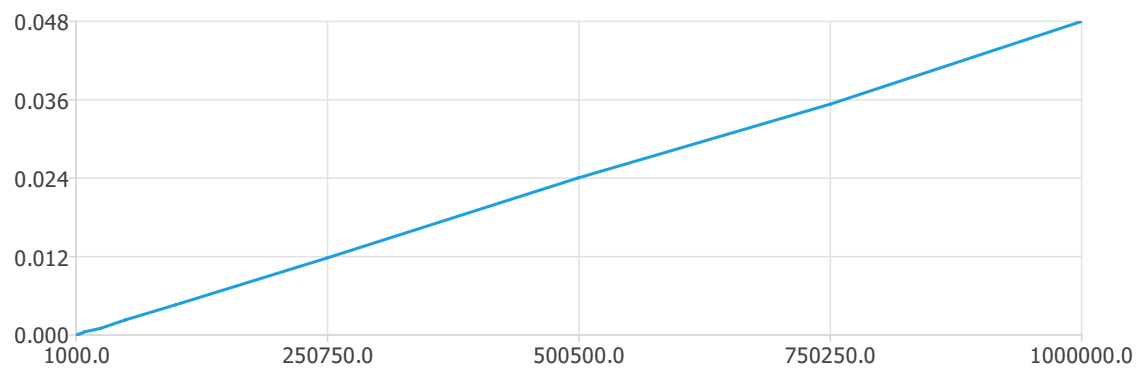
Quick Hull

Figure 19: generování

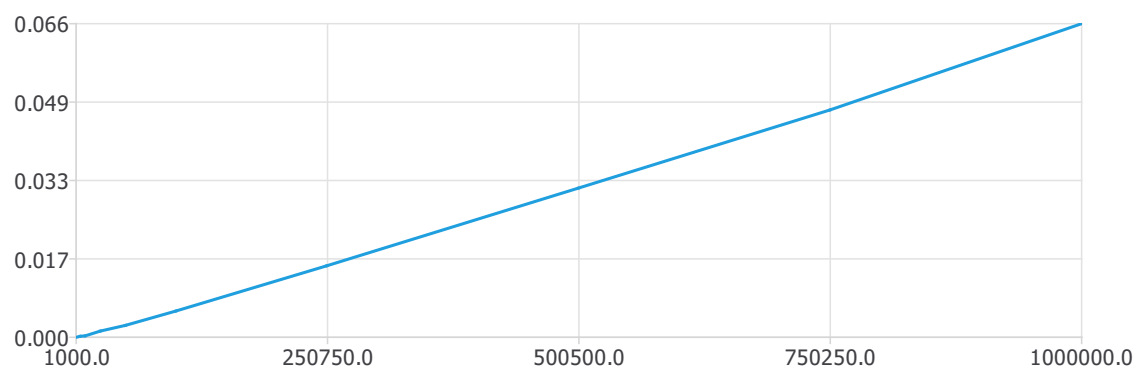


Figure 20: generování 10x

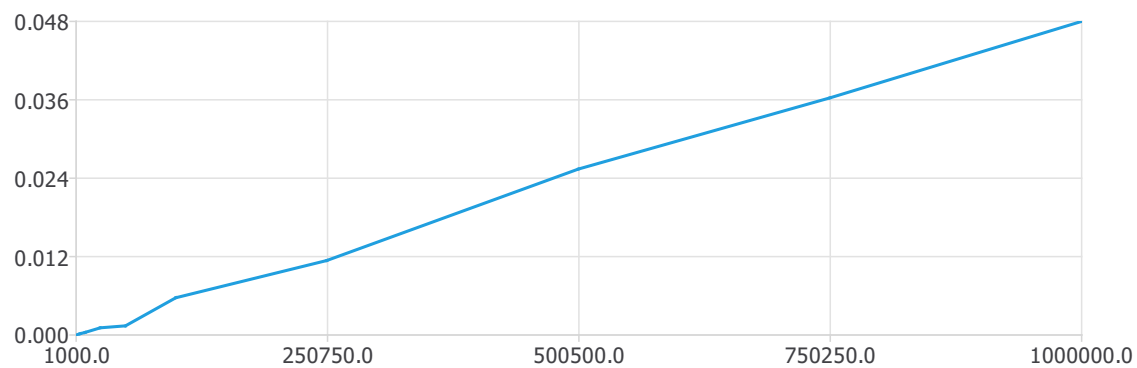
Incremental construction

Figure 21: generování

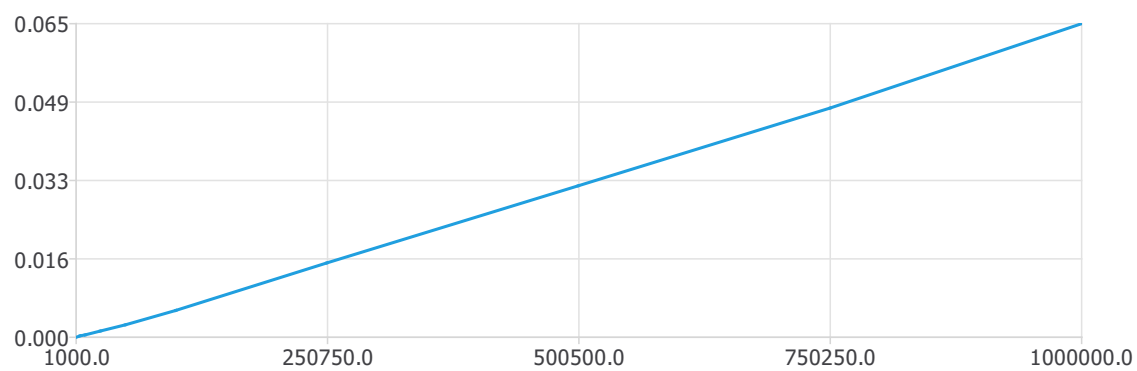


Figure 22: generování 10x

7 Závěr

Autoři splnili většinu bodů zadání a vznikl program, který generuje soubor bodů, jejich rozmístění a počítá rychlost výpočtu algoritmu. V předchozí kapitole jsou zobrazeny grafy, které znázorňují průběhy výpočtu algoritmů. Test byl proveden několikrát. Jak je vidět z grafů, nejvíce nestabilní je Jarvis Scan. Dochází k velkému kolísání v počátečních aj dokonce u středních hodnot. O něco lépe je na tom Quick Hull, a však aj u něho dochází ke značným výchylům. Obecně dalo by se říci, že Incremental construction je nejvíce 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 klesá zároveň výchylka mezi jednotlivými testy. Zároveň s rostoucím počtem n klesá rozdíl mezi jednotlivými metodami. Z výsledku lze usoudit, že

pro generování menšího objemu dat je lepší využít Incremental construction, případně Quick Hall, naopak pro velký objem dat co převyšuje velikosti desítek tisíc je možné využívat jakýkoliv algoritmus. Bohužel z časových důvodů nebylo možné splnit bonusové úlohy, proto třeba Graham Scan zůstal jen rozdělaný.

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.

V algoritmu Incremental pro malé hodnoty (cca 10 bodů) dochází k tomu, že občas některý bod není zahrnut

do vzniku nové obálky. Bohužel chybu nešlo přímo detekovat, protože má nahodilý charakter.

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 opět transformace.