Západočeská univerzita v Plzni

Fakulta aplikovaných věd

Bakalářská práce

Správa a vizualizace časoprostorových bodových dat

Plzeň, 2013 **Jindra Marvalová**

Prohlášení

Předkládám tímto k posouzení a následné obhajobě bakalářskou práci zpracovanou na závěr bakalářského studia na Fakultě aplikovaných věd Západočeské univerzity v Plzni. Prohlašuji, že jsem práci vypracovala samostatně pod odborným vedením vedoucího práce a s použitím literatury a pramenů, jejichž úplný seznam je uveden na konci práce.

V Plzni dne 30. května2012 ……

Jindra Marvalová

Poděkování

Na tomto místě bych ráda poděkovala vedoucímu bakalářské práce panu Ing. Janu Ježkovi, Ph.D. za téma práce, odborné vedení celé práce, inspirativní rady a připomínky.

Abstrakt

Tato bakalářská práce se zabývá kartografickou generalizací bodových dat. Jsou zde navrženy a popsány čtyři metody použitelné ke shlukování bodů, tyto metody jsou testovány na bodech hry geocaching v ČR a jsou mezi sebou porovnány z hlediska vhodnosti metod pro konkrétní použitá data. Dále jsou zde popsány možnosti kartografické vizualizace generalizovaných dat a navrženy tři způsoby vizualizace, provedené v programu ArcGIS.

Klíčová slova

Časoprostorová bodová data; Generalizace; Shlukování; k-means; SnapToGrid; Geohash; MarkerClusterer; Vizualizace

Abstract

This thesis deals with the cartographic generalization of point data. There are designed and described four clustering methods. These methods are tested on points of geocaching in the Czech Republic and they are compared with each other in terms of the appropriateness for given data. Additionally, there are described possibilities of cartographic visualization of generalized data and there are suggested three ways of visualization in ArcGIS.

Key words

Spatio-temporal point data; Generalization of point data; Clustering; k-means; SnapToGrid; Geohash; MarkerClusterer; Visualization

Obsah

Seznam obrázků

Seznam tabulek

1 Úvod

Název práce, správa a vizualizace časoprostorových bodových dat, vymezuje poměrně široké téma. V průběhu práce byla zaměřena pozornost na dílčí problém, efektivní generalizaci dat a jejich správu v databázi. Co se týče správy dat, je práce zaměřena především na zachování konzistence mezi původními daty a daty generalizovanými a na algoritmizaci procesu kartografické generalizace.

Cílem této bakalářské práce je tvorba relační databáze pro uchování dat, návrh vhodné metody pro kartografickou generalizaci bodových dat a následnou vizualizaci generalizovaných dat a otestování navržených metod na reálných datech.

Teoretická část práce se zabývá kartografickou generalizací a vizualizací. Generalizace znamená v kartografii proces zjednodušení obsahu mapy. Jedná se o proces řízené redukce objemu dat, při němž dochází k nevratné ztrátě informací. Klasický přístup ke kartografické generalizaci obnáší manuální práci kartografa, který vytváří generalizovanou mapu pro jednotlivá měřítka. Je to proces poměrně pomalý, velice subjektivní a jeho výsledek je závislý na osobě kartografa. Informační technologie postupně umožňují proces generalizace algoritmizovat, vzhledem ke složitosti tohoto problému je ale algoritmizace obtížná. Existuje velké množství metod, ty se však velmi liší svojí efektivitou a výslednými výstupy. Problém kartografické generalizace je velice aktuální, v loňském roce se jím zabývala diplomová práce [1] na Západočeské univerzitě.

Tato bakalářská práce navrhuje a porovnává možnosti automatické generalizace bodových dat. K řešení tohoto problému jsou zde navrženy a podrobně popsány čtyři metody, metoda k-means, SnapToGrid, Geohash a MarkerClusterer. V teoretické části práce jsou dále popsány faktory a metody kartografické generalizace a metody kartografické vizualizace, které jsou vhodné k zobrazení bodových dat a jejich shluků. Uvedené shlukovací metody jsou v praktické části práce testovány na reálných datech a pro generalizovaná data jsou navrženy tři způsoby kartografické vizualizace.

Teoretická část práce navazuje především na kvalifikační práce studentů [2], [3], [4], vědecké články [5], [6]. Dále pak na odbornou kartografickou literaturu [11], [12] a další zdroje.

2 Shluková analýza

Shluková analýza je souhrnný název pro všechny výpočetní postupy, které seskupují různé objekty do skupin tak, že podobnost dvou objektů náležejících do jedné skupiny je maximální, zatímco podobnost s objekty mimo tuto skupinu je minimální. Shluk (cluster) je pak skupina objektů, které jsou si navzájem podobné, ale zároveň rozdílné od ostatních objektů, které do skupiny nepatří. Prostorové shlukování je seskupování objektů do skupin na základě jejich vzdálenosti nebo jejich relativní hustoty v prostoru.

Shlukování se obecně používá k řešení mnoha problémů. Například k digitálnímu zpracování obrazu, v ekonomii se dá využít pro třídění zákazníků, ve statistice se při velkém počtu prvků používají jen středy shluků, které zastupují celý shluk. Dále se využívá pro vyhledávání v textových dokumentech nebo pro data mining (česky vytěžováni, dolováni dat). Velice specifické uplatnění našlo shlukování i v biologii, například při vytváření skupin genů, s podobnými modely chování.

Problém shlukování je velice aktuální a to především z důvodu neustále se zvyšujícího objemu dat. Tímto problémem se aktuálně zabývá několik projektů, je publikováno mnoho článků a zabývají se jím i závěrečné práce studentů.

Shlukování (clustering) bývá někdy zaměňováno s klasifikací (clasification). Termín „class“ bývá někdy používán jako synonymum k termínu „cluster“. Mezi těmito dvěma přístupy je ale rozdíl. Při klasifikaci jsou prvky rozdělovány do předem daných tříd, zatímco při shlukování tyto třídy neznáme, můžeme znát maximálně jejich počet.

V této kapitole jsou metody shlukové analýzy rozděleny na hierarchické a nehierarchické a některé metody jsou podrobně popsány. Kapitola vychází především z [4], [5] a [6].

2.1 Metody shlukové analýzy

Shlukovací metody se dají rozdělit do dvou rozdílných kategorií na hierarchické a nehierarchické. Toto rozdělení je podrobněji popsáno v následujících podkapitolách. Dále je zde popsáno několik metod, který je možné použít ke shlukování bodových dat,

a na konci kapitoly je tabulka 2.1, ve které jsou popsaným metodám přehledně přiřazeny jejich vlastnosti.

2.1.1 Hierarchické metody

Hierarchické metody využívají shluků, které byly nalezeny již dříve, a vytváří z nich shluky nové. Tím vzniká hierarchická stromová struktura. Průnikem každých dvou podmnožin je při hierarchickém shlukování buď jedna z nich, nebo prázdná množina. Tyto metody se dají rozdělit do dvou základních podskupin na aglomerativní a divizní.

Při aglomerativním přístupu se vyberou dva nejpodobnější jednoprvkové objekty a spojí se do jednoho shluku. Tento shluk je pak zařazen jako celek a postup se opakuje tak dlouho, dokud všechny objekty netvoří jeden shluk nebo nevznikne předem definovaný počet shluků.

Divizní metody berou naopak množinu všech objektů, které chceme shlukovat jako jeden celek a jeho postupným dělením vzniká hierarchický systém shluků. Dělení probíhá tak dlouho, dokud nejsou všechny shluky jednoprvkové. [5]

2.1.2 Nehierarchické metody

Tyto metody nevytvářejí hierarchickou strukturu. Rozloží dané objekty do skupin dle předem daného kritéria a první rozklad se dále nedělí, ale optimalizuje se vzdálenost a odlišnost shluků. Před započtením analýzy je nutné najít optimální počet shluků. Podle toho můžeme tyto metody dělit na metody s počtem shluků, který je předem pevně daný, a na metody, které počet shluků mění za běhu.

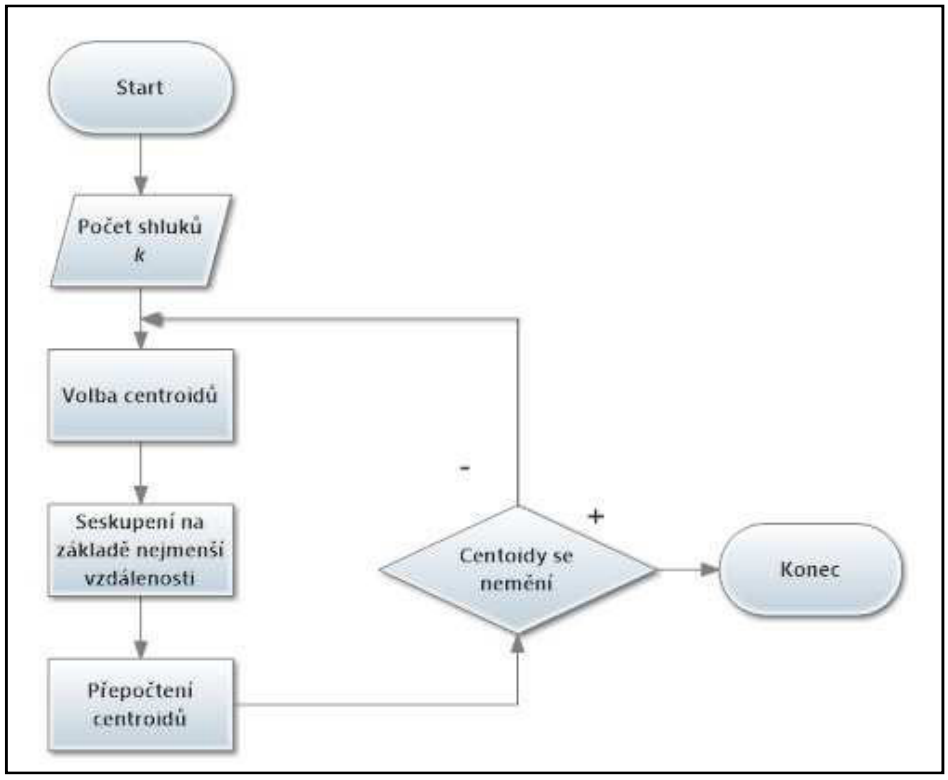
2.2 Metoda K-means

Metoda K-means, česky také K-průměrová metoda, je poměrně jednoduchá a velmi rozšířená metoda. Vznikla roku 1967, jejím autorem je J. B. MacQueen. Na této metodě se zakládá velké množství dalších algoritmů, které ji modifikují nebo ji přímo využívají.

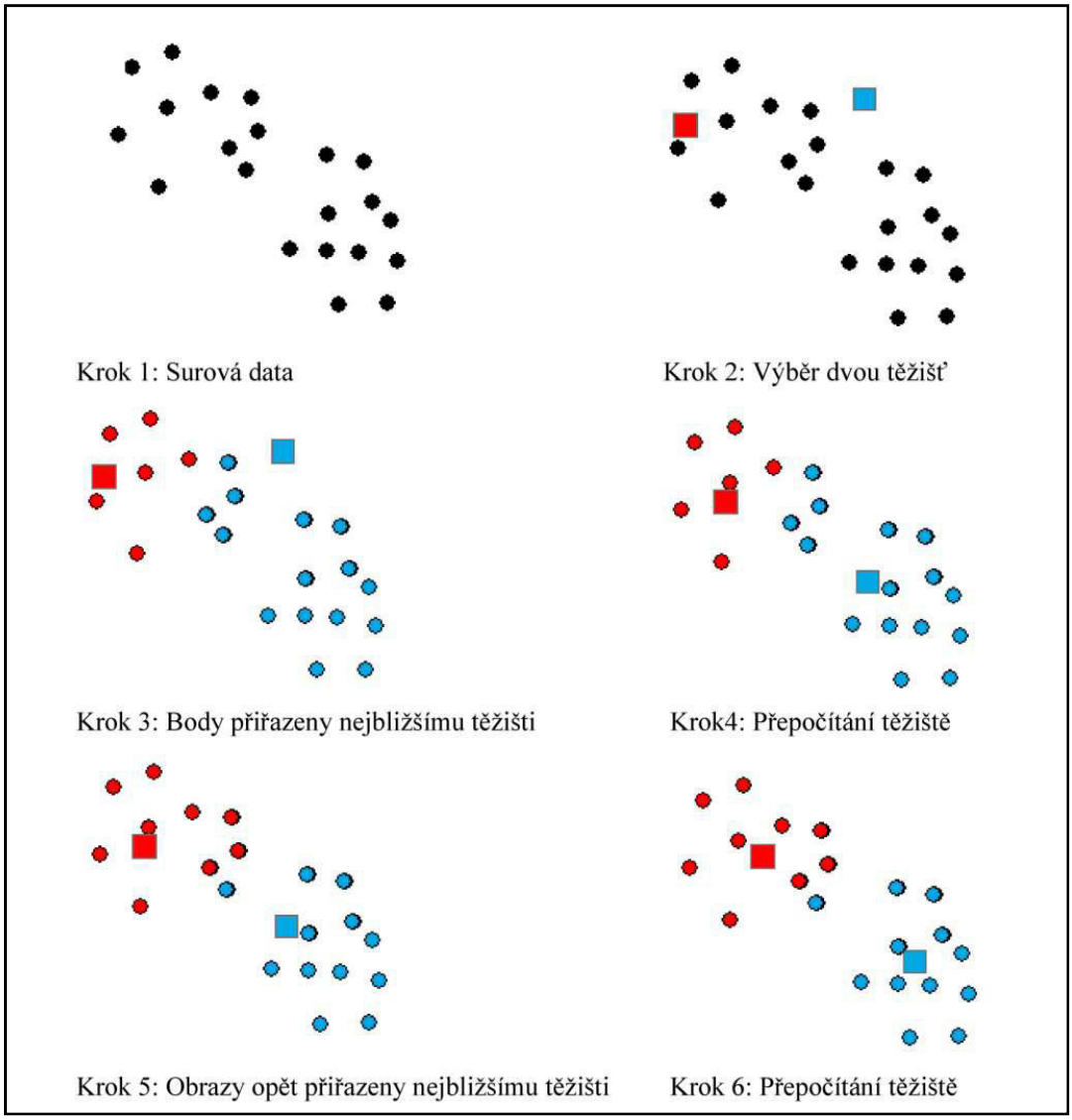
Metoda využívá jednoduchý způsob, jak rozdělit data pomocí předem definovaného počtu shluků. Hlavní myšlenkou metody je nějakým způsobem definovat *k* těžišť (centroidů) každého shluku. Dalším krokem je spojit každý jednotlivý bod s nejbližším těžištěm. Pokud takto spojíme všechny body, je první seskupování hotové. Po tomto seskupení se přepočtou těžiště. Jednotlivé body se opět spojí s nejbližšími těžišti. Tím vzniká cyklus, ve kterém v každém kroku všech *k* těžišť mnění své umístění. Cyklus je ukončen ve chvíli, kdy přestanou probíhat změny, jinými slovy ve chvíli, kdy se těžiště přestanou hýbat. Obecně se dá postup metody popsat čtyřmi kroky:

1. Urči *k* bodů a označ je jako těžiště shluků.
2. Každý bod přiřaď nejbližšímu těžišti. Body přiřazené jednomu těžišti označ za shluk.
3. Pro každý shluk najdi nové těžiště.
4. Kroky 2 a 3 se opakuj, dokud se těžiště nepřestanou pohybovat.

Tento postup je znázorněn na vývojovém diagramu na obrázku 2.1. Ukázka průběhu dvou iterací metody pro *k*=2 je znázorněna na obrázku 2.2.



Obr. 2.1: Vývojový diagram metody k-means



Obr. 2.2: Průběh metody k-means

Největšími výhodami této metody jsou její jednoduchost, a také to, že konverguje v konečném počtu kroků. Nevýhodou je pak to, že je potřeba předem znát počet shluků. Další nevýhodou je to, že může existovat více řešení, s ohledem na počáteční podmínky. [5][6]

Existuje mnoho variant metody k-means, příkladem může být sférický k-means, fuzzy k-means, k-medoids a další. Tyto metody jsou popsány v následující podkapitole, která vychází především z [4].

2.2.1 Některé modifikace metody k-means

Optimalizované k-means

Optimalizované k-means je velmi rozšířenou podobou k-means. Na rozdíl od klasické k-means metody aktualizuje střed těžiště shluku již ve fázi přiřazování bodů. Každý jednotlivý bod se tedy před přiřazením porovnává s aktualizovaným těžištěm. Postup se dá popsat třemi body:

1. Urči *k* bodů a označ je jako těžiště shluků.
2. Každý bod přiřaď nejbližšímu těžišti. Po přiřazení každého jednotlivého bodu aktualizuj těžiště.
3. Pokud není splněna zastavovací podmínka, pokračuj bodem 2.

Ve většině případů přináší tato modifikace lepší výsledky, ale za cenu složitějšího výpočtu.

Metoda k-medoids

Metodě k-means je velmi podobná i metoda k-mediods. Medoid má podobný význam jako centroid, jen je vždy reprezentován reálným prvkem z datového souboru. Z toho vyplývá rozdíl této metody. Ten je pouze v tom, že jako těžiště shluků nejsou vytvářeny nové body, ale jsou za ně považovány existující body z datového souboru, které jsou reálnému těžišti nejblíže.

Metoda Fuzzy k-means

Ve fuzzy k-means se pro každý prvek počítá, s jakou pravděpodobností patří do jakého shluku. Pomocí této metody lze dobře popsat rozložení bodů ve shlucích. Body na okrajích shluku totiž mají nižší stupeň příslušnosti ke shluku než body v blízkosti jeho středu. Jeden prvek může patřit i do více shluků zároveň. Příslušnost prvku ke shluku se vyjadřuje koeficientem příslušnosti. Součet všech koeficientů příslušnosti pro jeden prvek se musí rovnat jedné. Postup metody:

1. Vyber počet shluků.
2. Náhodně přiřaď koeficient příslušnosti ke každému bodu.
3. Opakuj následující kroky, dokud změna koeficientu příslušnosti není menší, než nějaký daný práh citlivosti a pak skonči.

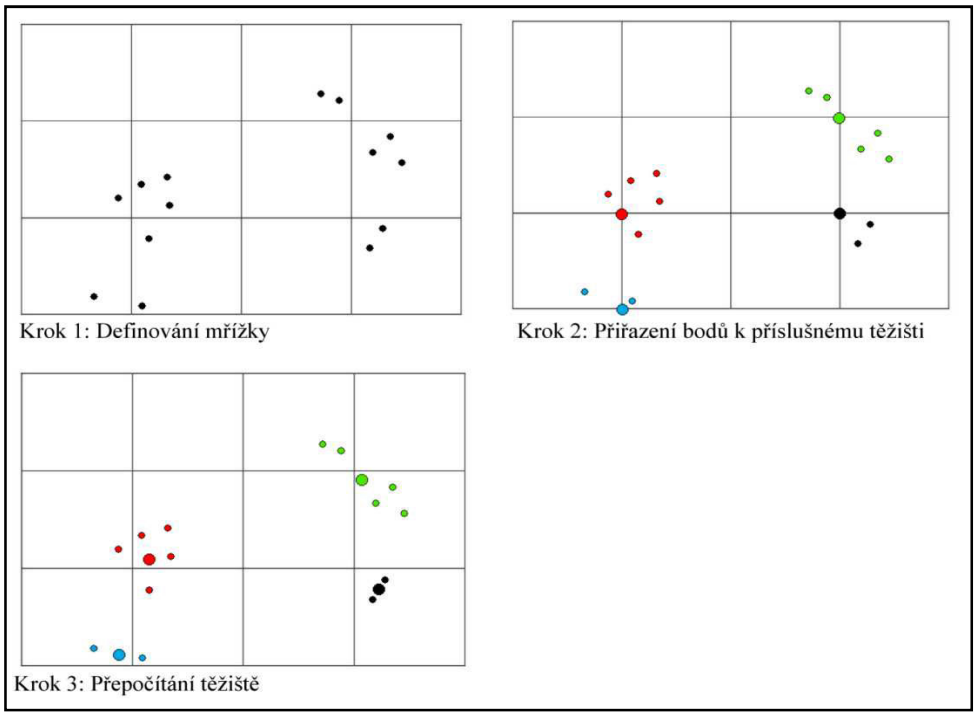
a) Spočítej střed každého shluku.
b) Pro každý bod spočítej pravděpodobnost příslušnosti k určitému shluku.

Sférická k-means

Při této variantě metody k-means se začíná se všemi prvky v jedné skupině. Tato skupina se pomocí klasické k-means rozdělí na dvě. Dále se rozdělí vždy jen jedna ze dvou vzniklých množin. K dalšímu dělení se obvykle vybírá ta s větším počtem prvků. Celý proces se opakuje a algoritmus končí ve chvíli, kdy dosáhne předem daného počtu shluků.

2.3 Metoda SnapToGrid

SnapToGrid je jedna z nejjednodušších metod používaných ke shlukování. Její princip je v rozdělení plochy pravidelnou mřížkou, která je dána počátečním bodem a velikostí buňky. Za těžiště shluků jsou považovány jednotlivé body mřížky. Prvky se přiřadí vždy k nejbližšímu těžišti. Pro účely shlukování je vhodné nakonec přepočítat polohu těžiště v rámci jednotlivých shluků. Postup metody při vytváření shluků je znázorněn na obrázku 2.3.

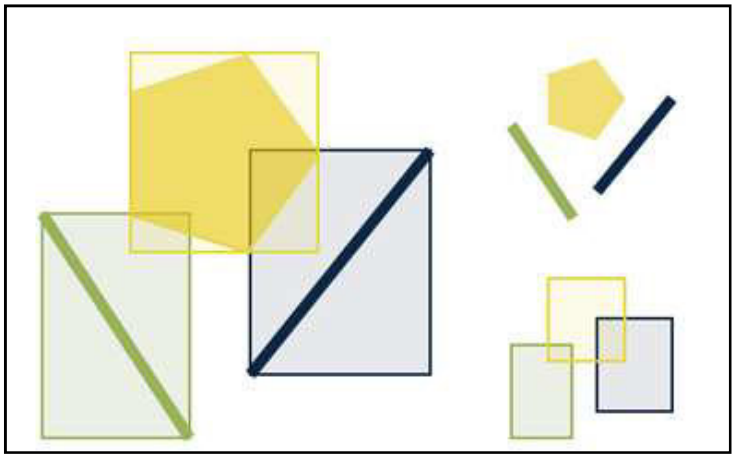


Obr. 2.3: Metoda SnapToGrid

2.4 Shlukování dat pomocí prostorového indexu

Prostorový index je jednou z největších výhod prostorové databáze. Umožňují používat databáze pro velké množství dat. Index je databázová konstrukce, která primárně slouží ke zrychlení vyhledávání a dotazování v databázi.

Tímto indexem se neoznačují jednotlivé prvky, ale index funguje tak, že se okolo každého prvku vytvoří tzv. bounding box a ten je označen indexem. Bounding box je nejmenším obdélníkem, rovnoběžným se souřadnicovými osami, který obsahuje daný prvek. Příklad bounding boxu je na obrázku 2.4. [7]



Obr. 2.4: Bounding boxy, zdroj [7]

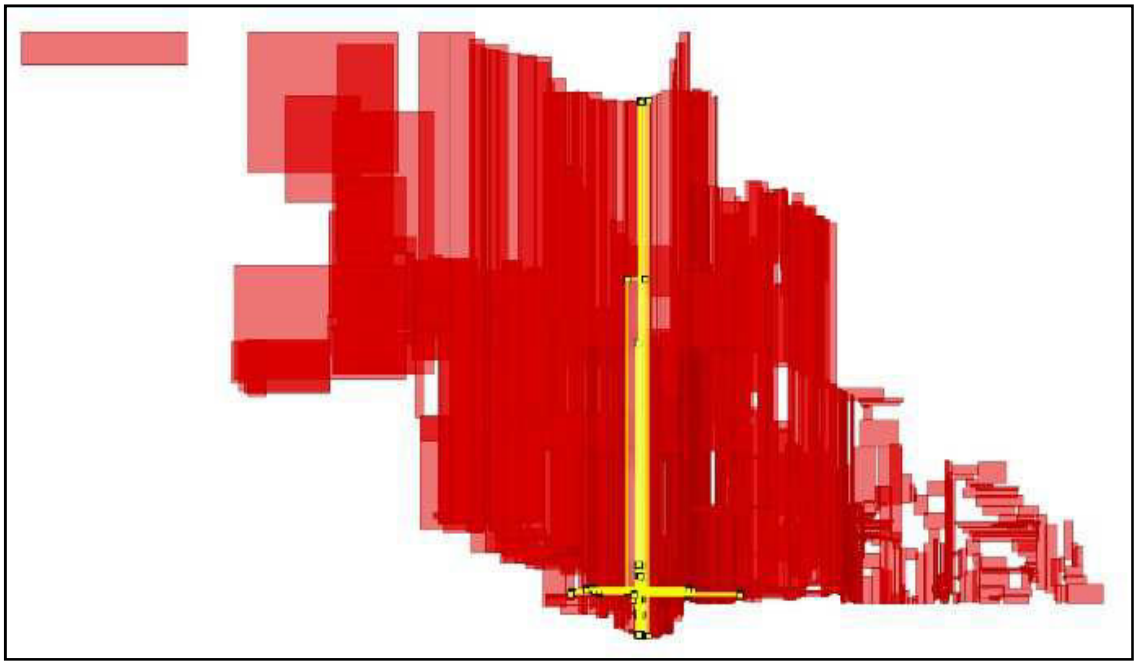
Bounding boxy se používají z toho důvodu, že i ty nejsložitější linie a polygony mohou být reprezentovány jednoduchým obdélníkem. Dotaz, zda polygon **A** leží uvnitř polygonu **B**, který je pro obecné polygony výpočetně velmi složitý, je tak možné převést na dotaz, zda bounding box **A** leží uvnitř bounding boxu **B**. Tyto prostorové indexy tedy pracují rychle, ale poskytují jen přibližné výsledky.

Prostorové indexy, realizované různými databázemi se velice liší. Nejběžnější je použití R-tree, který používá i PostGIS. Jiné prostorové databáze používají Quadtree a indexy založené na GRID. [7]

2.4.1 R-tree

R-strom vychází původně z B-stromu, ale liší se tím, že pro indexaci využívá vícerozměrnou informaci (pro geografická data dvourozměrnou). Každý uzel R-stromu obsahuje odkazy na několik potomků, jejichž počet by se měl pohybovat mezi *m* a *M*, pro které platí $m<M/2$. Uzly, které nejsou listy, obsahují ještě bounding box všech svých potomků. Oblasti jednotlivých uzlů se mohou překrývat.

Zajímavé je, že R-strom, vytvořený postupně na prostorových datech nemusí mít logickou strukturu. Ukázkou je obrázek 2.5, na kterém je zobrazen prostorový index listů na datech silnic v Britské Kolumbii. Je zřejmé, že jednotlivé bounding boxy jsou velice protáhlé a úzké. Velice rozdílná je i velikost jednotlivých bounding boxů. Z toho jasně vyplývá, že tato struktura není vhodná ke shlukování bodových dat. Další nevýhodou pro shlukování je již zmíněné překrývání bounding boxů. [7]



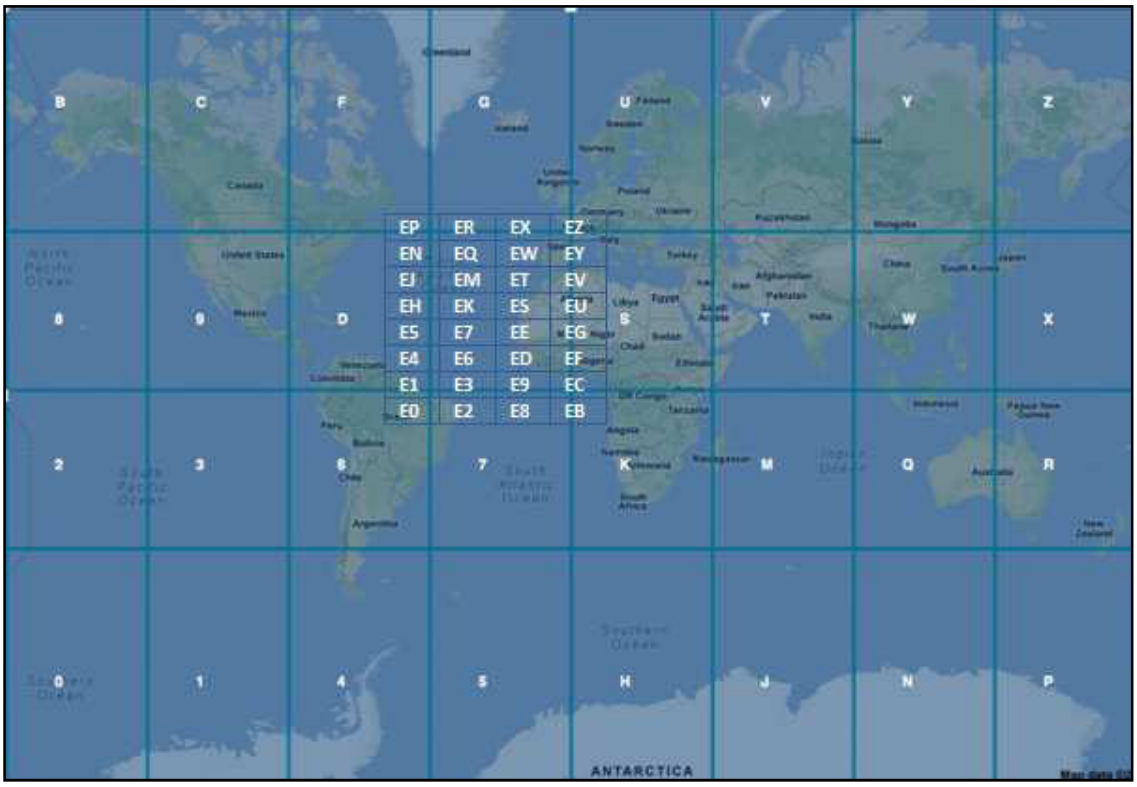
Obr. 2.5: R-strom, zdroj [7]

2.5 Metoda Geohash

Metoda Geohash vytváří systém kódů na základě zeměpisné šířky a délky. Dělením prostoru pomocí mřížky vytváří hierarchickou stromovou strukturu.

Funguje tak, že zemský povrch v zobrazení UTM rozdělí na 4 řádky a 8 sloupců a každou z těchto 32 buněk označí alfanumerickým znakem. Tímto znakem jsou číslice 0-9 nebo písmena B-H, J, K, M, N, P-Z. Takto vzniklá buňka se opět dělí na 32 buněk, tentokrát na 8 řádek a 4 sloupce, každá z nově vzniklých buněk se označí alfanumerickým znakem. Při dalším dělení vznikají opět 4 řádky a 8 sloupců, každá nová buňka se opět označí alfanumerickým znakem. Dělení je znázorněno na obrázku 2.6. Tímto postupným dělením prostoru a označováním jednotlivých vzniklých buněk se tvoří řetězec znaků, kód. [8]

Základní vlastností kódu je tedy to, že čím je kód delší, tím větší přesnost udává. Přesnost se tedy může jednoduše snižovat zkracováním kódu, to znamená ubíráním znaků na konci kódu. Platí také to, že čím delší je společná předpona v kódech dvou bodů, tím blíže by si tyto body měly být.



Obr. 2.6: Metoda Geohash

2.5 Metoda MarkerClusterer

Metodou MarkerClusterer nazývám metodu, kterou používá knihovna MarkerClustererPlus. MarkerClustererPlus pro Google maps v3 je pomocná klientská knihovna, která je součástí Google maps API. Umožňuje organizaci velkého množství bodů do shluků na základě jejich vzdálenosti od středu shluku.

MarkerClusterer funguje tak, že vezme první bod, určí ho za těžiště shluku a vytvoří okolo něj čtvercový bounding box. Pro další body pokračuje tak, že pokud se nenacházejí v bounding boxu žádného shluku, tvoří těžiště nového shluku. Pokud se nacházejí v bounding boxu některého shluku, přiřadí se do tohoto shluku. Pokud se nacházejí v bounding boxu více shluků, přiřadí se k tomu shluku, k jehož těžišti je nejblíže. Pokud je bod přiřazen k existujícímu shluku, přepočte se vždy těžiště shluku. Takto se postupuje, dokud všechny body nejsou přiřazeny do shluků. [9]

Výhodou je, že knihovna současně se shlukováním bodů řeší i vizualizaci. Shluky zobrazuje na mapách Google maps. Bounding boxy se volí vhodně velké a to vždy tak, aby se jednotlivé shluky při vizualizaci nepřekrývaly.

V tabulce 2.1 je uvedeno porovnání některých uvedených metod na základě několika hledisek. Znaménko **+** znamená, že metoda danou vlastnost splňuje, **-** značí opak.

Tab. 2.1: Porovnání metod

Vlastnost	k-means	fuzzy k-means	SnapToGrid	Geohash	MarkerClusterer
Počet shluků musíme určit předem	+	+	-	-	-
Závisí na počátečních podmínkách	+	+	-	-	-
Jeden prvek může patřit do více shluků	-	+	-	-	-
Hierarchická metoda	-	-	-	+	-

3 Kartografická generalizace

Z kartografického hlediska je shlukování bodů vlastně generalizací. Existuje mnoho definic kartografické generalizace, jednou z nich je i definice podle ČSN 73 046, která zní: „Kartografická generalizace spočívá ve výběru, geometrickém zjednodušení a zevšeobecnění objektů, jevů a jejich vzájemných vztahů pro jejich grafické vyjádření v mapě, ovlivněné účelem, měřítkem mapy a vlastním předmětem kartografického znázorňování.“ Dalším příkladem je definice Konstantina Alexejeviče Sališčeva, významného ruského kartografa a profesora Lomonosovy univerzity: „Kartografická generalizace je proces výběru a zevšeobecňování obsahu mapy mající na zřeteli zobrazení skutečností v jejich hlavních rysech a zvláštnostech podle účelu a měřítka mapy.“

K procesu generalizace existují dva přístupy. Prvním je tzv. geoprostorová generalizace, při které přechodem z většího měřítka do menšího dochází pouze ke zjednodušování tvaru prvků a neřeší se vztahy mezi jednotlivými prvky mapy. Druhým přístupem kartografická generalizace, která řeší konflikty mezi prvky mapy, aby nedocházelo ke ztrátě souvislostí.

Existuje několik metod kartografické vizualizace a faktorů, které ovlivňují její postup postup. Ty to metody a faktory jsou stručně popsány v další části této kapitoly.

3.1 Metody kartografické generalizace

Metody kartografické generalizace se dělí do několika skupin:

Generalizace výběrem - výběr prvků, které mají být z mapy odstraněny, nebo které mají být na základě svých charakteristik zvýrazněny.

Geometrická generalizace - úprava tvaru prvků na základě jejich geometrických vlastností.

Generalizace reklasifikací - slučování prvků podobných vlastností do tříd.

Generalizace agregací - Seskupování malých prvků podobných vlastností tak, aby mohly být reprezentovány jedním prvkem.

Generalizace prostorovou redukcí - změna dimenze prvku.

Generalizace ploch - základní operace s plochami, jako např. sjednocení ploch, rozdělení ploch nebo zrušení ploch.

Generalizace atributů - výběr atributů, které zůstanou zachovány, na základě předem daných kritérií.

Jednotlivé metody jsou blíže popsány například v [10]. Tyto generalizační metody jsou vzájemně provázané a většinou nebývá na data aplikována pouze jedna metoda, ale jejich kombinace. Důležitou roli hraje také pořadí, v jakém jsou jednotlivé metody na data aplikovány.

3.2 Faktory kartografické generalizace

Při výběru a aplikaci metod kartografické generalizace musí být brány v úvahu následující faktory:

Účel mapy - správná mapa by měla zdůraznit ty prvky, které jsou důležité s ohledem na účel mapy, a měla by je upřednostňovat před prvky nedůležitými vzhledem k účelu.

Měřítko mapy - větší měřítko mapy dovoluje větší podrobnost, to znamená menší míru generalizace.

Charakteristiky území - generalizaci je nutné realizovat s ohledem na konkrétní území. Generalizace bude vypadat úplně jinak ve venkovských oblastech a v oblastech městských, kde je potřeba zobrazit více rozdílných informací.

Kvalita dat - při zpracování dat je třeba brát ohled na jejich kvalitu. Ta může být velice rozdílná s ohledem na to, z jakých zdrojů data pocházejí.

Kartografické vyjadřovací prostředky – generalizaci je nutné provádět s ohledem na použité kartografické vyjadřovací prostředky. Ty totiž velmi ovlivňují přehlednost a čitelnost výsledného kartografického produktu.

Nejdůležitějšími faktory pro kartografickou generalizaci jsou měřítko mapy a účel mapy. Měřítko mapy je velice důležité pro výsledný vzhled mapy a její vypovídací hodnotu. Měřítko určuje podrobnost mapy a tedy i míru generalizace. Větší měřítko umožňuje menší míru generalizace, ale za cenu menší rozlohy zobrazeného území. Naopak malá měřítka umožňují zobrazení velkého území, ale obsah mapy je značně generalizován. Podle účelu mapy se pak určují prvky, které jsou na mapě důležité a prvky, které je naopak možno potlačit.

3.3 Automatizace kartografické generalizace

Automatizace kartografické generalizace je aktuálním problémem, který úzce souvisí především s rychle rostoucím objemem geodat a s neustále se zvyšující produkcí mapových výstupů. V současné době ale zatím neexistuje mezi GIS produkty žádný, který by dokázal komplexně řešit problematiku kartografické generalizace. Řešení tohoto problému tedy zůstává z velké části na kartografech. [11]

4 Metody kartografické vizualizace

Pojem vizualizace znamená reprezentaci reálného jevu, stavu nebo dat do obrazové podoby analogové nebo digitální. Její nenahraditelnou výhodou je názornost a schopnost předávat informace srozumitelnou formou. Kartografická vizualizace je reprezentace dat formou mapy.

Při výběru vhodných metod, vhodných ke kartografické generalizaci shluků, je potřeba brát v úvahu dvě charakteristiky shluku, počet bodů shluku a území, ze kterého se body seskupily do jednoho shluku. Počet bodů shluku je veličina s velkým rozsahem. Jednotlivé shluky mohou být jednobodové, nebo mohou nabývat až tisíce prvků.

Podrobný popis metod kartografické vizualizace je možné najít například v [11] nebo [12]. Především z této literatury vychází i následující popis metody teček, metody bodových znaků a metody plošných znaků, které jsou vhodné k vizualizaci bodových dat a jejich shluků.

4.1 Metoda teček

Metoda teček, jež je někdy nazývaná tečkovou metodou, původně vznikla pro vyjádření hustoty zalidnění. Dnes se používá k vyjadřování diskrétních charakteristik s nerovnoměrnou distribucí, například hustoty zalidnění, těžby nerostných surovin nebo pěstování zemědělských plodin. Metoda slouží pouze k reprezentaci prostorového rozmístění jevu, neslouží k reprezentaci absolutních hodnot jevu, protože teček je ve většině případů takové množství, že není možné je spočítat. Základním kartografickým vyjadřovacím znakem této metody jsou tečky tak malých rozměrů, že je lze zanedbat. Metoda funguje tak, že v místě velké koncentrace jevu dochází ke slévání teček, naopak v místech malé koncentrace jevu jsou jednotlivé tečky samostatné a řídce rozmístěné.

V závislosti na vyjadřovaném jevu mohou být tečky polohově lokalizované, čímž je znázorněna kromě distribuce jevu i lokalizované umístění, nebo mohou být na daném území rozmístěny plošným způsobem. Při plošném rozložení jsou tečky rovnoměrně rozloženy po celé ploše územních celků v pravidelné mřížce. Tento způsob rozmístění teček je obdobou kartogramu.

Kvantitativní hodnotu zobrazovaného jevu je možné vyjádřit pomocí váhy tečky. Váha tečky je dána přiřazeném konkrétní kvantitativní hodnoty k velikosti tečky. Různou barvou, rastrem nebo geometrickým tvarem tečky je možné v mapě vyjádřit kvalitativní charakter dat nebo více jevů zároveň.

Dalším důležitým aspektem metody je velikost tečky. Ta musí být určena velice pečlivě, jinak mapa nemá potřebnou vypovídací hodnotu. Při volbě příliš malé velikosti tečky nebude metoda správně zobrazovat plošné rozložení jevu, rozložení se bude zdát příliš řídké. Naopak při volbě příliš velké tečky se budou tečky v místě velké koncentrace jevu slévat a vyvoláme tak dojem celkově vysoké koncentrace jevu.

Reprezentace bodů metodou teček je vhodná pro vizualizaci plošného rozložení bodů v malém měřítku. Tato vizualizace je dále v práci využita a zobrazena na obrázku 5.3.

4.2 Metoda bodových znaků

Metoda bodových znaků slouží k interpretaci bodových jevů nebo plošných jevů, které není možné v měřítku mapy zobrazit plochou. Základním vyjadřovacím prostředkem metody jsou, jak již říká název, bodové znaky. Výběr znaků je velice zajímavým kartografickým úkolem. Symbolů je totiž nepřeberné množství, ale je jen velice málo pravidel pro jejich výběr.

Na metodu bodových znaků lze pohlížet dvěma způsoby a to z hlediska parametru bodového znaku. Metodou je možné vyjadřovat kvantitativní i kvalitativní charakter jevu.

4.2.1 Reprezentace kvalitativních jevů

Kvalitativní jevy se vyjadřují tvarem, strukturou, výplní a orientací bodového znaku. Podle tvaru se bodové znaky rozdělí na geometrické, symbolické, alfanumerické a obrázkové. Geometrické znaky jsou jednoduché geometrické tvary, např. kruh, trojúhelník, čtverec. Symbolické znaky jsou jednoduché kresby typových objektů a prvků, např. kostel, rozhledna, letadlo. Obrázkové znaky pak představují kresby konkrétních a jedinečných objektů, např. Eiffelova věž nebo Tower Bridge. Obrázkový znak se tedy na mapě nemůže opakovat. Výhodou obrázkových a symbolických znaků

je snadná čitelnost, názornost. Nevýhodou je jejich grafická složitost a velikost. Alfanumerickými znaky jsou písmena a číslice. Tyto znaky musí být používány s rozvahou tak, aby nemohly být v mapě zaměněny s popisky. Tvar znaku se většinou volí podle tvaru zobrazovaného objektu. Studna tedy bude reprezentována kruhovou značkou, budova čtvercovou nebo obdélníkovou.

Strukturou znaku se rozumí vnitřní grafické členění znaku. Struktura je důležitá především k reprezentaci vnitřní struktury jevů. Využívá se především při tvorbě kartodiagramů. Výplň znaku může být dána tónem barvy nebo texturou znaku nebo jeho části. Barvy a textura znaku se volí buď asociativně, funkčně (např. tak, aby spolu nesousedily znaky podobných barev) nebo mohou být vybírány podle estetického hlediska a zvyšovat designovou atraktivitu mapy.

Orientací znaku se rozumí natočení znaku kolem jeho osy nebo těžiště. Nejčastěji se používá pro znázornění směru jevu (migrace ptáků, směr větru) nebo k orientaci znaků podle souřadnicové sítě.

4.2.2 Reprezentace kvantitativních jevů

Kvantitativní jevy se vyjadřují velikostí znaku. Velikost znaku může někdy odpovídat pouze významu nebo důležitosti zobrazovaného jevu, pokud má ale velikost znaku odpovídat skuteční hodnotě, je potřeba vytvořit stupnici a vypočítat velikost znaku pro každou hodnotu jevu, popřípadě pro určitý interval. Velikost znaku může být přímo i nepřímo úměrná kvantitě zobrazovaného jevu.

Metoda bodových znaků je vhodná k bodové reprezentaci těžišť shluků pro jakékoli měřítko mapy a k reprezentaci jednotlivých bodů v měřítku větším než 1:100 000. Výhodou reprezentace shluků bodovými znaky je menší grafické zatížení mapy. Nevýhodou je to, že tato reprezentace neposkytne žádnou informaci o prostoru, ze kterého se body seskupily do jednoho shluku.

4.3 Metoda plošných znaků

Z geometrického hlediska mají téměř všechny existující objekty plošný charakter. Vlivem měřítka mapy se ale mnoho plošných objektů na mapě zobrazuje bodovými nebo liniovými znaky. Pouze ty jevy, které lze v měřítku mapy zakreslit

plošně, se znázorňují metodou plošných znaků. Metoda je založena na používání plošných znaků. Je součástí i jiných kartografických metod, např. kartogramu, dasymetrické metody nebo se používá k vyjadřování izoplet. Z hlediska prostorového uspořádání znaků se plošné znaky dělí na izolované, dotykové a prolínající se znaky.

Na rozdíl od metody bodových znaků využívá metoda plošných znaků jen dva parametry, obrys a výplň. Výplň znaku se využívá k vyjádření kvalitativních i kvantitativních charakteristik jevu, obrys především k vyjádření kvalitativních charakteristik.

Výplň plošného znaku představuje zaplnění ohraničené plochy znaku barvou, liniovým nebo bodovým rastrem, bodovými znaky nebo popisem. Výplň není volena náhodně, ale řídí se mnoha kartografickými pravidly. Co se týče reprezentace kvalitativního jevu, platí, že podobné jevy se znázorňují podobnými barvami (rastry) a naopak různé jevy se znázorňují různými barvami (rastry). Barvy i rastry se volí asociativně, např. les zeleně, okresy s převažující většinou voličů ODS modře atd.

Při reprezentaci kvantitativních jevů platí, že čím vyšší je intenzita jevu, tím vyšší je intenzita rastru. Pro znázornění intenzity rastru se volí změna tloušťky nebo hustoty, ale ne změna směru rastru. Totéž platí i při použití barev. Čím vyšší je intenzita zobrazovaného jevu, tím tmavší je použitá barva.

Obrys znaku je tvořen linií, která ohraničuje výplň plochy. Tato linie má všechny parametry liniových znaků – tloušťku, barvu, strukturu. Z hlediska ohraničující linie se dají plošné znaky kategorizovat do čtyř skupin na znaky ohraničené, otevřené, neúplně ohraničené a dynamické. Tloušťka linie se volí podle důležitosti a hierarchické nadřazenosti areálu. Čím důležitější je areál, tím silnější linie je pro obrys volena. Čárkované a tečkované linie se volí pro ohraničení areálů, jejichž hranice se nedá přesně určit, nebo pro areály s přibližným výskytem jevu.

Tato metoda je vhodná pro zobrazení shluků v libovolném měřítku. Shluky budou reprezentovány plochou, ze které se body seskupily do shluku. Výhodou je názornější reprezentace shluků, nevýhodou je větší grafické zatížení mapy a nemožnost reprezentovat těžiště shluků.

5 Datový model

V zadání bakalářské práce je dáno, že navržené metody mají být otestovány na datech bodového pole v Nečtinech. Po analýze dostupných dat bylo ale zjištěno, že těchto bodů je dost málo, navíc jsou v prostoru rozmístěny poměrně pravidelně. To znamená, že jejich charakter není vhodný pro testování shlukovacích metod. Pro tato data byl tedy vytvořen datový model, který byl inspirován modelem vytvořeným v bakalářské práci [13]. Tento model zde nebude uveden ani podrobněji pospán z toho důvodu, že tato data nebyla pro testování použita.

Pro testování bylo tedy nutné najít jiná, vhodnější data. Podmínkou pro nová data bylo, aby se jednalo o bodová data prostorově lokalizovaná a nerovnoměrně rozmístěná. Počet bodů se měl pohybovat v řádu tisíců. Naskytla se možnost získat data dopravních nehod v Libereckém kraji, ta by pravděpodobně byla pro testování ideální. Data se ale nakonec získat nepodařilo. Pro testování byla tedy vybrána data hry geocaching na území České republiky. Jedná se o prostorově lokalizovaná bodová data, která jsou dána souřadnicemi GPS a datem vzniku.

Nejstarší bod vznikl v roce 2001, nejnovější v roce 2013. Data byla stahována začátkem roku 2013, dnes tedy již nejsou úplně aktuální, protože nové body vznikají v podstatě každý den.

5.2 Databáze Geocaching

Geocaching je celosvětová hra na pomezí sportu a turistiky, která využívá GPS zařízení k hledání ukrytých schránek. Hráči pomocí zadaných souřadnic a GPS přístroje hledají místo, kde se ukrývá hledaná schránka (keš). Keše jsou ukryty po celém světě, většinou na zajímavých nebo neobvyklých místech.

5.2.1 Stažení dat

Informace o jednotlivých bodech byly staženy ze serveru [14]. Zde je možné keše vyhledávat podle adresy, země, poštovního směrovacího čísla, podle zeměpisné

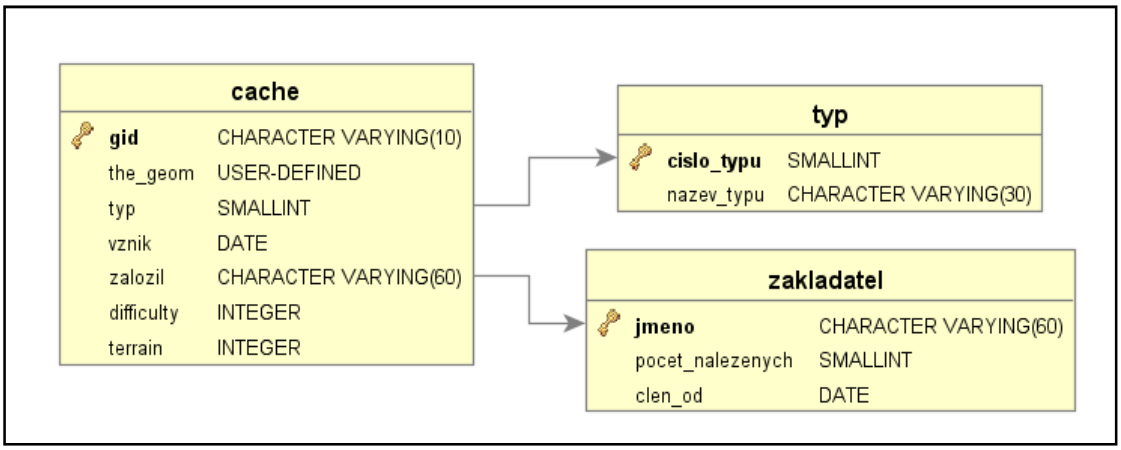
šířky a délky a dalších kritérii. Já jsem zvolila vyhledávání podle země a následně podle jednotlivých krajů České republiky.

Data je možné stáhnout po souhlasu s Groundspeak licenční smlouvou. Úplná licenční smlouva je dostupná na adrese [14].

Stažení dat v takovém rozsahu je možné pomocí premium účtu, jehož založení je zpoplatněno symbolickou částkou, nebo existují i jiné alternativní postupy. Data se stahují ve formátu GPX. GPX (the GPS Exchange Format) je datový formát s open licencí. Jedná se o textový soubor s XML strukturou, který se používá k výměně mezi zařízeními GPS a v různých aplikacích.

5.2.2 Datový model Geocaching

Na obrázku 5.1 je zobrazen datový model, který slouží k uložení dat.



Obr. 5.1: E-R-A model databáze Geocaching

Databáze je tvořena třemi entitami: ***cache***, ***typ***, ***zakladatel***.

Tabulka ***cache*** obsahuje informace o keších:

gid unikátní číslo s předponou GC, které jednoznačně označuje keš

the_geom souřadnice ve formě geometrie PostGIS

typ číslo 1-7, které značí typ keše

vznik datum založení keše ve formátu RRRR-MM-DD

zalozil přezdívka, kterou zakladatel keše používá na webu [14],

difficulty celé číslo 1-5, které udává, jak snadné je objevit keš

terrain celé číslo 1-5, které vystihuje náročnost terénu při hledání keše.

Tabulka ***zakladatel***:

jmeno přezdívka, kterou zakladatel keše používá na webu [14]

pocet_nalezenych počet keší, které nalezl

clen_od datum, od kdy je zaregistrován na webu [14], ve formátu RRRR-MM-DD.

Tabulka ***typ***:

cislo_typu celé číslo 1-7, které označuje typ keše

nazev_typu typ keše (tradiční, multikeš, mystery, letterbox, earth keš, webcam keš, event keš).

Více o jednotlivých typech je možné zjistit v [14].

5.3 Vytvoření databázového systému v PostGIS

Pro správu dat se využívá databázový systém. Databázový systém je tvořen databází a systémem řízení báze dat (DBMS z anglického database management system). DBMS je počítačový program pro organizaci a správu databáze. Jeho úkolem je rychle poskytovat data uživatelům a zároveň zajistit integritu dat. Chrání data proti smazání nebo poškozená a usnadňuje doplnění, odstranění nebo aktualizaci dat. [15]

S daty pracuji v DBMS PostgreSQL s nadstavbou PostGIS.

5.3.1 DBMS PostgreSQL a PostGIS

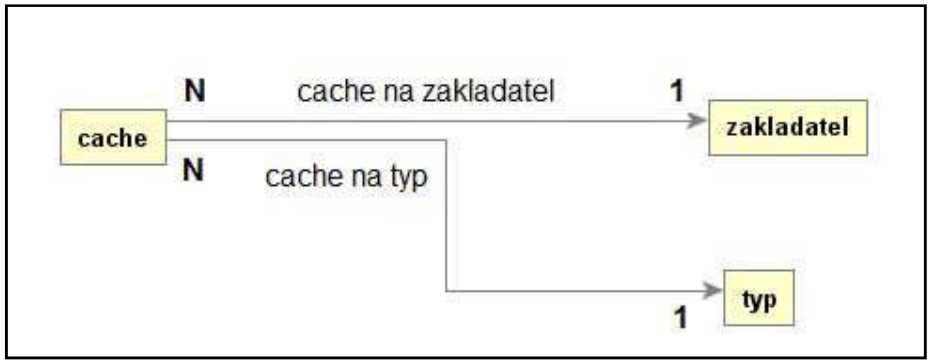
PostgreSQL, často zjednodušeně Postgres, je objektově-relační DBMS. Jedná se o nejvyspělejší existující open source systém, který prošel více než patnácti lety vývoje, a který svojí rychlostí a funkčností může konkurovat komerčním databázím. Disponuje mnoha funkcemi, které se v jiných databázových systémech objevují málokdy, některé dokonce vůbec. [16]

PostGIS je prostorová nadstavba k objektově-relačnímu systému PostgreSQL. Jedná se o rozšíření pro geografická data. Je vyvíjen společností Refraction Research Inc. a je stále zdokonalován. Nabízí mnoho prostorových operátorů, prostorové funkce a prostorové datové typy, funguje na většině operačních systémů. [16]

PostGIS se řídí normou „Simple Features Specification for SQL“. Standard je velice obsáhlý a je k dispozici na webových stránkách Open Geospatial Consortium, Inc. [17]

5.3.2 Struktura datového modelu v PostGIS

Datový model je tvořen tabulkami a vazbami mezi nimi. Ve sloupcích tabulek jsou uloženy jednotlivé atributy s přiřazeným datovým typem. Model databáze s vazbami je znázorněn na obrázku 5.2. Tabulky ***cache*** a ***zakladatel*** jsou spojeny vazbou ‚*cache na zakladatel*‘, která je typu N:1. To znamená, že jeden zakladatel mohl založit více keší, ale jedna keš má pouze jediného zakladatele. Tabulky ***cache*** a ***typ*** jsou spojeny vazbou ‚*cache na typ*‘, která je typu N:1.



Obr. 5.2: Model databáze Geocaching - vazby

5.3.3 Práce s databází v aplikaci PgADMIN

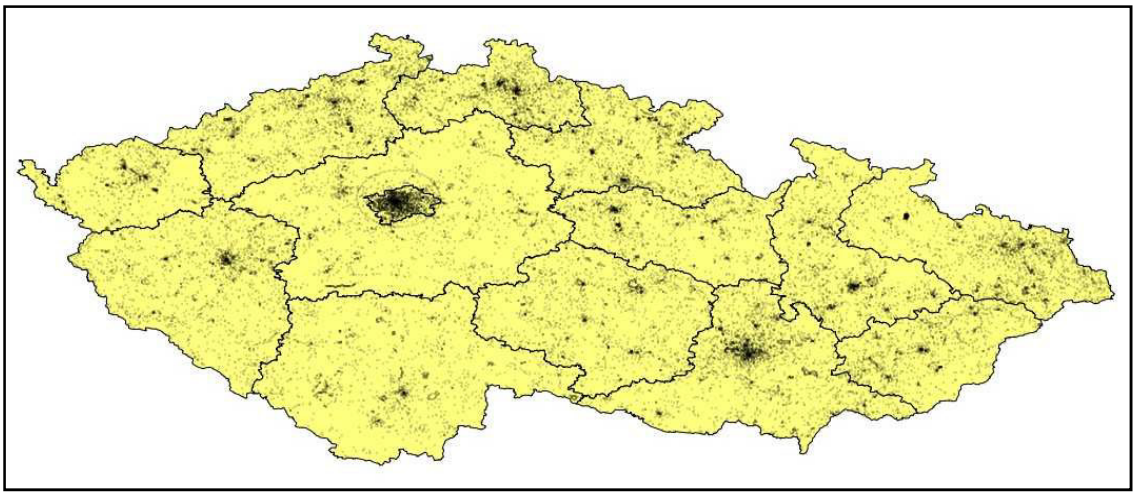
Pro vytvoření databázových tabulek byl použit populární open source databázový nástroj PgAdmin, určený k administraci databází PostgreSQL. Nabízí grafické rozhraní, podporující všechny funkce PstgreSQL, editor SQL se zvýrazněním syntaxe, je lokalizován do mnoha jazyků a portován pro celou řadu operačních systémů.

Vytvoření tabulek spočívá ve vytvoření a spuštění SQL skriptů. Skripty mohou být napsány v editoru SQL nebo mohou být vytvořeny pomocí uživatelského rozhraní kliknutím na *Tabulky* → *Nová tabulka* a vyplněním požadovaných parametrů tabulky. Tím se SQL skript vygeneruje automaticky. Ukázka skriptu je uvedena zde, všechny ostatní skripty, použité k vytvoření tabulek, jsou uvedeny v příloze.

```
CREATE TABLE cache
(gid character varying(10) NOT NULL,
  the_geom geometry,
  typ smallint,
  vznik date,
  zalozil character varying(60),
  difficulty integer,
  terrain integer,
  CONSTRAINT "identifikator bodu" PRIMARY KEY (gid),
  CONSTRAINT "cash na typ" FOREIGN KEY (typ)
      REFERENCES typ (cislo_typu) MATCH SIMPLE
      ON UPDATE NO ACTION ON DELETE NO ACTION,
  CONSTRAINT "cash na zakladatel" FOREIGN KEY (zalozil)
      REFERENCES zakladatel (jmeno) MATCH SIMPLE
      ON UPDATE NO ACTION ON DELETE NO ACTION)
```

5.4 Naplnění databáze

Data byla do databáze vložena ze souborů GPX pomocí skriptu, napsaného v jazyce PHP. PHP je programovací jazyk fungující na straně serveru, který byl vyvinut především pro tvorbu internetových stránek a webových aplikací. Celý skript je uveden v příloze. Do databáze bylo vloženo 28 142 bodů, 9750 zakladatelů a 7 typů keší.



Obr. 5.3: Distribuce bodů

Po uložení bodů do databáze bylo možné data zobrazit metodou teček. Z obrázku 5.3 je zřejmé, že vysoká koncentrace bodů je ve městech a v oblasti severních Čech a ve východním cípu České republiky. Naopak nižší hustota bodů je ve venkovských oblastech, v oblasti jižních Čech a v oblastech vojenských újezdů se téměř žádné body nenacházejí.

6 Realizace shluků

Jak již bylo řečeno, při generalizaci dat je nutné brát úvahu faktory kartografické generalizace. Pro body geocachingu je to především měřítko mapy a použité kartografické vyjadřovací prostředky. Při zobrazení bodů v malém měřítku je potřeba body seskupit do velkých shluků, naopak při zobrazení ve velkém měřítku je možné zobrazit body jednotlivě, případně zobrazit popisky bodů.

Vzhledem k následné kartografické vizualizaci shluků je potřeba brát v úvahu i kartografické vyjadřovací prostředky. K počtu shluků je potřeba zvolit vhodnou velikost kartografických značek tak, aby se značky nepřekrývaly.

V tabulce 6.1 jsou uvedeny navrhované počty shluků pro jednotlivá měřítka, velikost jednoho kilometru pro dané měřítko a počet pixelů, na něž se zobrazí, při velikosti pixelu 0,25 mm.

Tab. 6.1: Navrhované počty shluků pro jednotlivá měřítka

Měřítko mapy	Velikost km v měřítku mapy [cm]	Počet pixelů, na které se zobrazí jeden km	Navrhovaný počet shluků
1 : 1 500 000	0,07	3	30
1 : 1 000 000	0,1	4	50
1 : 500 000	0,2	8	150
1 : 300 000	0,3	13	300
1 : 200 000	0,5	20	500
1 : 100 000	1	40	1100

Tato kapitola popisuje realizaci shluků pomocí metod k-means, SnapToGrid a Geohash v PostGIS a realizaci shluků pomocí knihovny MarkerClusterer. Jednotlivé metody zde budou porovnány z hlediska vhodnosti použití jednotlivých metod pro daná data a vhodnosti pro následnou kartografickou vizualizaci.

6.1 Funkce k-means

Funkce k-means není součástí PostGIS. Hitoshi Harada vytvořil open source modul, jež implementuje funkci k-means do prostředí PostGIS, pod licencí PostgreSQL License. Tato funkce má tři parametry. Prvním parametrem je vstupní pole bodů, které se budou shlukovat, druhým parametrem je číslo *k* ve formátu integer, které udává počet shluků, které se mají vytvořit. Pro hodnoty 1, 0 a záporná čísla se vytvoří vždy jeden shluk. Třetí parametr, takzvané počáteční pole, je nepovinný. Jedná se o pole bodů, které budou považovány za počáteční těžiště shluků. Počet prvků tohoto pole musí být stejný jako počet shluků, který se vytváří. Pokud není třetí parametr zadán, zvolí k-means počáteční pole podle následujícího vzorce:

$$mean = (max - min) \cdot \frac{i}{k+1} + min,$$

kde *k* označuje počet shluků, *max* je pole, které obsahuje *k* nejvyšších hodnot shlukovaných dat, *min* je pole, které obsahuje *k* nejmenších hodnot shlukovaných dat a *i* nabývá hodnot od 1 do *k*.

Funkce k-means je v PostGIS volána jako agregační funkce. Agregační funkce umožňují pomocí příkazu `GROUP BY` seskupit vybrané prvky databáze a provádět nad nimi statistické nebo aritmetické výpočty. Ukázka SQL příkazu k vytvoření *k* shluků pomocí funkce kmeans:

```
CREATE TABLE table_name AS (SELECT kmeans, count(*),
ST_Centroid(ST_Collect(the_geom)) AS the_geom
FROM (SELECT kmeans(ARRAY[ST_X(the_geom), ST_Y(the_geom)], K)
OVER (), the_geom FROM cache) AS ksub
GROUP BY kmeans);
```

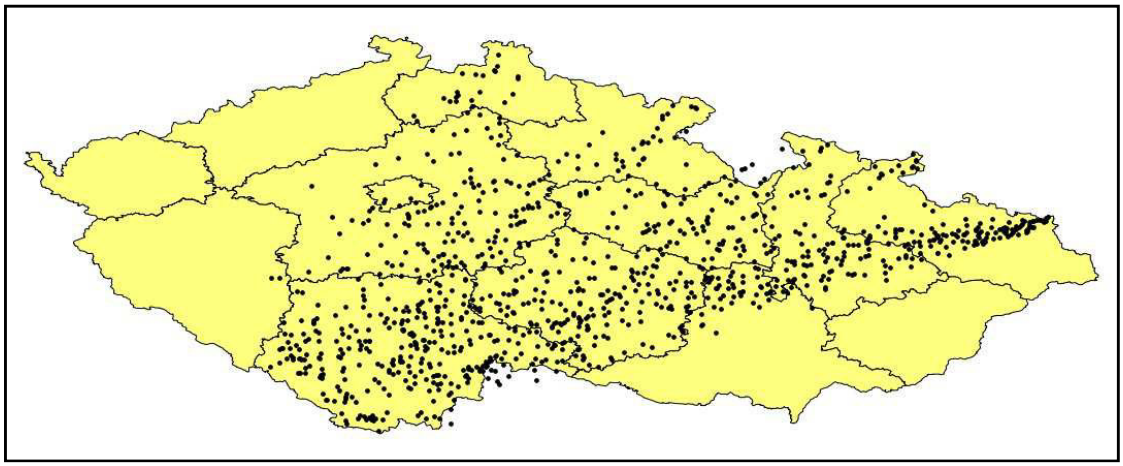
6.1.1 Funkce k-means bez použití třetího parametru

Funkcí k-means, bez použití třetího parametru, bylo v databázi vytvořeno šest tabulek, ve kterých jsou uloženy shluky. Každá tabulka má jiný počet shluků a je určena pro zobrazení v jiném měřítku. Přehled těchto tabulek je zobrazen v tabulce 6. 2.

Tab. 6.2: Shluky metodou k-means bez počátečního pole

Název tabulky	Vhodné měřítko	Počet shluků	Počet prvků ve shlucích
kmeans1500	1:1 500 000	30	355 - 2760
kmeans1000	1:1 000 000	50	76 - 2551
kmeans500	1:500 000	150	1 - 1318
kmeans300	1:300 000	300	1 - 841
kmeans200	1: 200 000	500	1 - 743
kmeans100	1:100 000	1100	1 - 633

Shluky vytvořené tímto způsobem se pro jednotlivá měřítka liší ne jen počtem vytvořených shluků, ale především rozložením shluků na území republiky. V měřítku 1:1 500 000 a 1:1 000 000 jsou shluky poměrně rovnoměrně rozloženy a plošná velikost shluků je srovnatelná. Ve větších měřítkách se ale charakter shluků mění. Distribuce shluků je velice nerovnoměrná. V oblasti Šumavy, Jeseníků a v pásmu napříč republikou od jihozápadu na severovýchod vzniká velké množství shluků, které jsou plošně velmi malé a ve většině případů obsahují jen několik málo bodů. V ostatních oblastech naopak vzniká jen malé množství velkých shluků, které obsahují výrazně vyšší počet bodů. To je způsobeno tím, jak funkce kmeans volí počáteční pole podle výše uvedeného vzorce. Počáteční vektor pro měřítko 1:100 000 je na obrázku Obr. 6.1. Podobně vypadají i počáteční pole pro ostatní měřítka.



Obr. 6.1: Počáteční pole funkce k-means

Výsledné shluky budí dojem, že v oblasti velké koncentrace shluků mají body výrazně odlišnou distribuci od zbytku území. Může se zdát, že v této oblasti je bodů méně a jsou od sebe vzdáleny. To ale skutečné distribuci bodů neodpovídá. Tyto shluky tedy nejsou pro reprezentaci dat příliš vhodné.

6.1.2 Funkce k-means s použitím třetího parametru

Jak již bylo řečeno v popisu funkce k-means, výsledky této funkce jsou závislé na počátečních podmínkách neboli počátečním poli vstupujícím do metody. Jako třetí parametr byla použita těžiště shluků, vytvořené metodou SnapToGrid. Ty jsou poměrně rovnoměrně rozmístěny po území republiky. Počet vytvářených shluků se tedy odvíjí od počtu bodů v počátečním poli. Počet vytvořených shluků tedy odpovídá počtu shluků vytvořených funkcí SnapToGrid. Vytvořené shluky jsou uvedeny v tabulce 6.3.

Tab. 6.3: Shluky metodou k-means s počátečním polem

Název tabulky	Vhodné měřítko	Počet shluků	Počet prvků ve shlucích
vstkm1500	1:1 500 000	29	449 - 2672
vstkm1000	1:1 000 000	46	231 - 2657
vstkm500	1:500 000	145	27- 1277
vstkm300	1:300 000	297	1 - 624
vstkm200	1: 200 000	501	3 - 461
vstkm100	1:100 000	1072	1 - 360

Takto vytvořené shluky jsou pro všechna měřítka rovnoměrně rozmístěné po území republiky a i plošná velikost shluků v jednotlivých měřítkách je srovnatelná. V měřítku 1:100 000 a 1:200 000 je již patrná vyšší koncentrace shluků v oblastech velkých měst jako Praha, Brno, Liberec, Plzeň nebo Ostrava.

6.2 Funkce SnapToGrid

Funkce SnapToGrid je součástí PostGIS. Tato funkce přichytí všechny dané body k pravidelné mřížce. Funkce má dva parametry. První parametr je počáteční bod

mřížky a druhý parametr je velikost buňky ve stupních. Funkce SnapToGrid je volána jako agregační funkce. Ukázka SQL příkazu k vytvoření shluků pomocí funkce SnapToGrid:

```
CREATE TABLE table_name AS (SELECT array_agg(gid) AS ids,
COUNT (the_geom ) AS count,
(ST_Centroid(ST_Collect( the_geom ))) AS center FROM cache
GROUP BY ST_SnapToGrid((the_geom), 0.1, 0.1));
```

Funkcí SnapToGrid bylo v databázi vytvořeno 6 tabulek, ve kterých jsou uloženy shluky. Každá tabulka je určena pro zobrazení v jiném měřítku a obsahuje jiný počet shluků. Jejich přehled je v tabulce 6.4.

Tab. 6.4: Shluky metodou SnapToGrid

Název tabulky	Vhodné měřítko	Velikost buňky	Počet shluků	Počet prvků ve shlucích
stg1500	1:1 500 000	0,8 · 0,8	29	3 - 3963
stg1000	1:1 000 000	0,6 · 0,6	46	3 - 2039
stg500	1:500 000	0,3 · 0,3	145	1 - 1934
stg300	1:300 000	0,2 · 0,2	297	1 - 1244
stg200	1: 200 000	0,15 · 0,15	501	1 - 1141
stg100	1:100 000	0,1 · 0,1	1072	1 - 736

Těžiště vzniklých shluků tvoří v měřítku 1:1 500 000 a 1:1 000 000 poměrně pravidelnou mřížku, od níž se výrazně odchyluje jen několik málo bodů, především na okrajích území. Plošná velikost shluků je uvnitř území poměrně stejná, velikostí se výrazně odlišují shluky na okrajích území. Totéž platí i pro větší měřítka jen s tím rozdílem, že v některých oblastech dochází k výraznějšímu narušení mřížky a v některých oblastech jsou již patrné rozdíly v plošné velikosti shluků.

6.3 Funkce Geohash

Funkce Geohash je stejně jako funkce SnapToGrid součástí PostGIS. Stejně jako předchozí funkce se v PostGIS volá jako agregační funkce. Jak již bylo řečeno, tato funkce postupně dělí prostor pravidelnou mřížkou a vytváří geohash kód. Počet vytvořených shluků je závislý na délce kódu. Počet vzniklých shluků pro některé délky kódu a vhodné měřítko jsou uvedeny v tabulce 6.5. Je zřejmé, že počty shluků se pro jednotlivé délky kódu rychle rostou. Funkce tedy pro shlukování poskytuje jen omezené možnosti. Zde je uvedena ukázka SQL kódu pro vytvoření shluků pomocí funkce Geohash při délce kódu *k*:

```
CREATE TABLE table_name AS (SELECT ST_geohash(the_geom, 5),
ST_Centroid(ST_Collect(the_geom)), count(*) the_geom
FROM cache
GROUP BY ST_geohash(the_geom, k));
```

Tab. 6.5: Shluky metodou Geohash

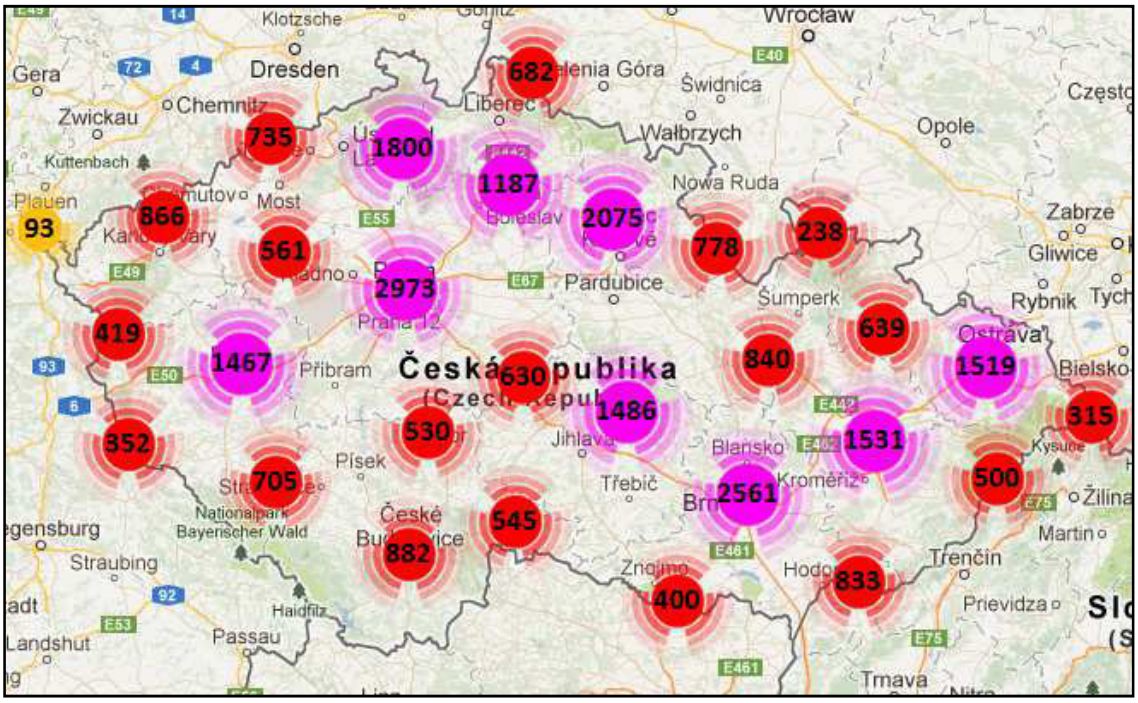
Název tabulky	Vhodné měřítko	Délka kódu	Počet shluků	Počet prvků ve shluku
geohash3	1:2 000 000	3	13	201-5738
geohash4	1:400 000	4	199	1-979
geohash5	1:40 000	5	4524	1-238

Shluky vytvořené funkcí Geohash o délce kódu 3 a 4 jsou rovnoměrně rozmístěny po území republiky. Plošná velikost shluků se výrazně odlišuje u shluků na okraji území. Shluky vytvořené funkcí Geohash o délce kódu 5 mají poměrně rozdílnou plošnou velikost, vzniká velké množství shluků, které obsahují jediný bod. Větší shluky jsou v oblastech velkých měst, jako Praha, Brno, Plzeň, v oblasti severních Čech a ve východním cípu republiky.

6.4 Použití knihovny MarkerClusterer

Knihovna MarkerClusterer je pro shlukování velice oblíbená díky jednoduchému použití. Další její výhodou je to, že spolu se shlukováním automaticky řeší i vizualizaci.

Skript, použitý k vytvoření shluků pomocí knihovny MarkerClusterer, je uveden v příloze. Knihovna funguje tak, že body z databáze načte a vytváření shluků probíhá přímo na internetové stránce, kde se shluky zobrazí. Vzniklé shluky na mapě v měřítku 1:3 000 000 jsou na obrázku Obr. 6.2. Tato knihovna byla vytvořena a je běžně používána pro generalizaci menšího počtu bodů, než pro jaký je použita zde. Vytvoření shluků tedy trvá poměrně dlouho.



Obr. 6.2: Shluky pomocí MarkerClusterer

6.4.1 Vizualizace shluků knihovnou MarkerClusterer

Jak již bylo řečeno, knihovna zároveň s vytvářením shluků řeší i jejich vizualizaci na mapě. Vizualizace na první pohled vypadá efektivně, má ale několik nevýhod a kartografických nedostatků. Shluky jsou reprezentovány znaky, které působí plošným dojmem. Uvnitř znaku je číslo, které udává počet bodů ve shluku. Pro znaky

platí kvantitativní pravidlo, že čím větší je shluk, tím větší je i znak, který ho reprezentuje. To je použito pravděpodobně i z toho důvodu, aby se čísla do znaků vešla a nepřesahovala. Velikost shluků je odlišena i jejich barvou. Platí, že shluky o velikosti 2-9 bodů mají barvu modrou, 10-99 žlutou, 100-999 červenou, 1000-9999 světle fialovou a shluky o velikosti 10 000 bodů a více mají tmavě fialovou barvu. Vzhledem k tomu, že velikost shluků (počet bodů ve shluku) je kvantitativní charakteristika, je použití kvalitativní barevné stupnice trochu neobvyklé. Znaky jsou také poměrně velké a kruhy, které tvoří střed znaku, jsou neprůhledné. Tím je každým znakem zakryta poměrně velká část mapy. Příkladem je shluk o velikosti 1467 bodů na obrázku 10, který na mapě úplně zakryje město Plzeň, popisek i okolí města.

Další nevýhodou této vizualizace jsou přesahy znaků přes hranice republiky. Mnoho znaků překrývá hranice a budí dojem, že body přiřazené do shluku leží i v místech, kde se žádné body nenacházejí. To je patrné například u shluku o velikosti 682 bodů severně od Liberce, na obrázku 6.2. Střed znaku, který by měl být totožný se středem shluku, i větší část znaku leží v Polsku, přitom všechny body, které tvoří shluk, leží v České republice.

6.5 Porovnání použitých metod

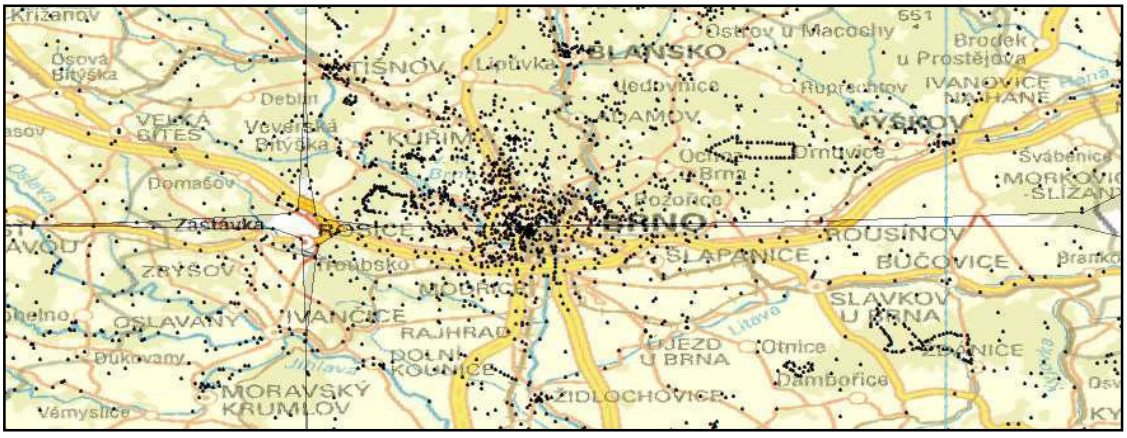
Metody shlukové analýzy je možné porovnávat z mnoha úhlů pohledu. V této práci jsou shluky vytvářeny za účelem kartografické generalizace a následné kartografické vizualizace. Hlavními kritérii pro hodnocení metod tedy bude to, jak shluky reprezentují daná data a především vhodnost vytvořených shluků pro kartografickou vizualizaci. Z hlediska kartografické generalizace je vhodné, aby metoda umožňovala určit nebo alespoň ovlivnit počet vytvořených shluků. Důležité je také to, zda metoda umožňuje vytvořit shluky pro libovolné měřítko mapy. Pro kartografickou vizualizaci je zase vhodné, aby se těžiště shluků nepřekrývala a aby plošná velikost shluků pro jednotlivá měřítka byla srovnatelná. Nemělo by tedy docházet k tomu, že při plošném zobrazení shluků bude jeden shluk velký a druhý tak malý, že se v měřítku mapy nezobrazí nebo bude v měřítku mapy velmi malý.

Výhodou metody k-means je to, že vytvoří libovolný počet shluků. Pomocí této metody tedy lze vytvářet shluky pro různá měřítka mapy. Nevýhodou jsou různé

výsledky metody pro různé počáteční podmínky. Je tedy nutné zvolit vhodné počáteční podmínky. Jak již bylo popsáno výše, shluky vytvořené metodou k-means bez použití počátečního pole reprezentují data velice nevhodně a takto vytvořené shluky nejsou vhodné ani ke kartografické vizualizaci, protože v některých místech vzniká velká koncentrace shluků a značky shluků by musely být reprezentovány velmi malými bodovými znaky, aby nedocházelo k jejich překrývání na mapě.

Metoda k-means s použitím počátečního pole vytváří shluky pravidelně rozložené po celém území, těžiště shluků se nepřekrývají a plošná velikost shluků je srovnatelná. Shluky data vhodně reprezentují, v oblastech s menším počtem bodů vzniká menší počet shluků, naopak v oblastech, kde se nachází více bodů, jsou shluky koncentrovanější a obsahují více bodů.

U metody SnapToGrid nemůžeme přesně určit počet shluků, které se mají vytvořit, tento počet se dá ale ovlivnit velikostí mřížky. Pro vytvoření většího počtu shluků musí mít mřížka menší velikost buňky. Shluky vytvořené touto metodou jsou rovnoměrně rozložené po celém území, ale plošná velikost znaků je velice rozdílná. Na okraji území vznikají často shluky, jež se při plošném zobrazení shluků v daném měřítku mapy nezobrazí nebo jsou velice malé. To je samozřejmě překážkou pro kartografickou vizualizaci pomocí metody plošných znaků.



Obr. 6.3: Shluky metodou SnapToGrid

Shluky navíc příliš nevystihují charakter dat. Dochází například k tomu, že body v jednom městě jsou rozděleny do dvou shluků. Tento případ je znázorněn na obrázku 6.3, kde jsou zobrazeny shluky vytvořené pro měřítko 1:1 5000 000. Město Brno je zde

v místě nejvyšší koncentrace bodů napříč rozděleno do dvou shluků (shluky jsou reprezentovány žlutými plochami). To je pochopitelné v případě, kdy se vytváří velké množství shluků a na ploše města by vzniklo třeba 5 shluků. V tomto případě je ale vytvářeno necelých 30 shluků a hranice dvou shluků leží v místě největší hustoty bodů.

Při vytváření shluků metodou Geohash můžeme počet vytvořených shluků ovlivnit pouze délkou kódu. Počet shluků se ale nemění plynule, ale skokově. Prosloužením kódu o jeden znak se počet shluků zvýší vždy mnohonásobně. Pomocí této metody tedy není možné vytvářet shluky pro libovolná měřítka. Možnosti metody pro shlukování jsou tedy omezené. Vytvořené shluky jsou poměrně rovnoměrně rozmístěny po celém území. Při délce kódu 3 vzniká pouze 13 shluků a nastává zde podobný problém jako u metody SnapToGrid. Příkladem je město Plzeň, body, které se zde nacházejí, jsou rozděleny do čtyř různých shluků. Při délce kódu 4 jsou těžiště shluků v pravidelné mříži, plošná velikost shluků je u všech shluků přibližně stejná, vymykají se pouze shluky na okraji území. Při délce kódu 5 vzniká již velké množství shluků, které věrohodně reprezentují zdrojová data. V oblastech velkých měst, na severu Čech a na východním okraji republiky, v oblastech vyšší hustoty bodů, vznikají větší shluky, naopak v ostatních oblastech převládají shluky tvořené jedním nebo dvěma body.

Z uvedeného přehledu vyplývá, že pro daná data je z použitých metod ke generalizaci nejvhodnější metoda k-means. Potvrdilo se ale to, že metoda je velmi citlivá na počáteční podmínky. Při volbě vhodných počátečních podmínek vzniknou shluky, které dobře reprezentují původní data. Výhody metody jsou zřejmé – možnost vytvoření libovolného počtu shluků, to znamená vytvoření shluků pro různá měřítka mapy, mezi plošnou velikostí shluků nejsou extrémní rozdíly. Navíc shluky vytvořené touto metodou nejlépe reprezentují daná data. Naopak shluky vzniklé metodami Geohash a SnapToGrid nereprezentují daná data příliš vhodně, jejich výhoda ale spočívá v tom, že jsou součástí PostGIS a jejich použití je velice snadné.

Uvedené charakteristiky jednotlivých metod jsou přehledně shrnuty v tabulce 6.6. Znaménko **+** vždy znamená, že metoda danou charakteristiku splňuje, znaménko **–** znamená opak.

Tab. 6.6: Porovnání použitých metod

Vlastnost	k-means	k-means s poč. polem	SnapToGrid	Geohash	MarkerClusterer
Vytvoření shluků pro libovolná měřítka	+	+	+	-	+
Možnost ovlivnit počet shluků	+	+	+	+	-
Vhodná reprezentace dat	-	+	-	-	+
Plošná velikost shluků je srovnatelná	-	+	-	-	+
Nedochází k překrytu těžišť shluků	-	+	+	+	+

7 Vizualizace shluků

Metody kartografické vizualizace, vhodné k zobrazení shluků, byly popsány v kapitole 4. Tato kapitola se zabývá konkrétní aplikací jednotlivých metod.

Pro kartografickou práci s daty byly v databázi vytvořeny shluky, reprezentované svým těžištěm, a shluky reprezentované plochou. Plocha jednotlivých shluků vznikla jako konvexní obálka všech bodů, které se přiřadí do shluku. Databáze se připojila k programu Quantum GIS (zkráceně QGIS), pomocí kterého byly databázové tabulky převedeny do formátu SHP. Další práce s pak probíhala v programu ArcGIS.

7.1 Vizualizace 1

Tato vizualizace byla inspirována vizualizací, kterou používá knihovna MarkerClusterer. Shluky jsou reprezentovány bodovými kruhovými znaky, v jejichž středu je číslem uveden počet bodů ve shluku. Znak je umístěn vždy v těžišti shluku.

Obr. 7.1: Vizualizace 1 – bodové znaky

Pomocí nepravidelné intervalové stupnice byly shluky rozděleny na 4 části. Pro jednotlivé intervaly je volena barva a velikost znaku tak, že čím více bodů je ve shluku, tím větší je znak, který shluk reprezentuje a tím tmavší je barva znaku. Stupnice, velikosti znaku i barvy jsou uvedeny v následující tabulce 7.1.

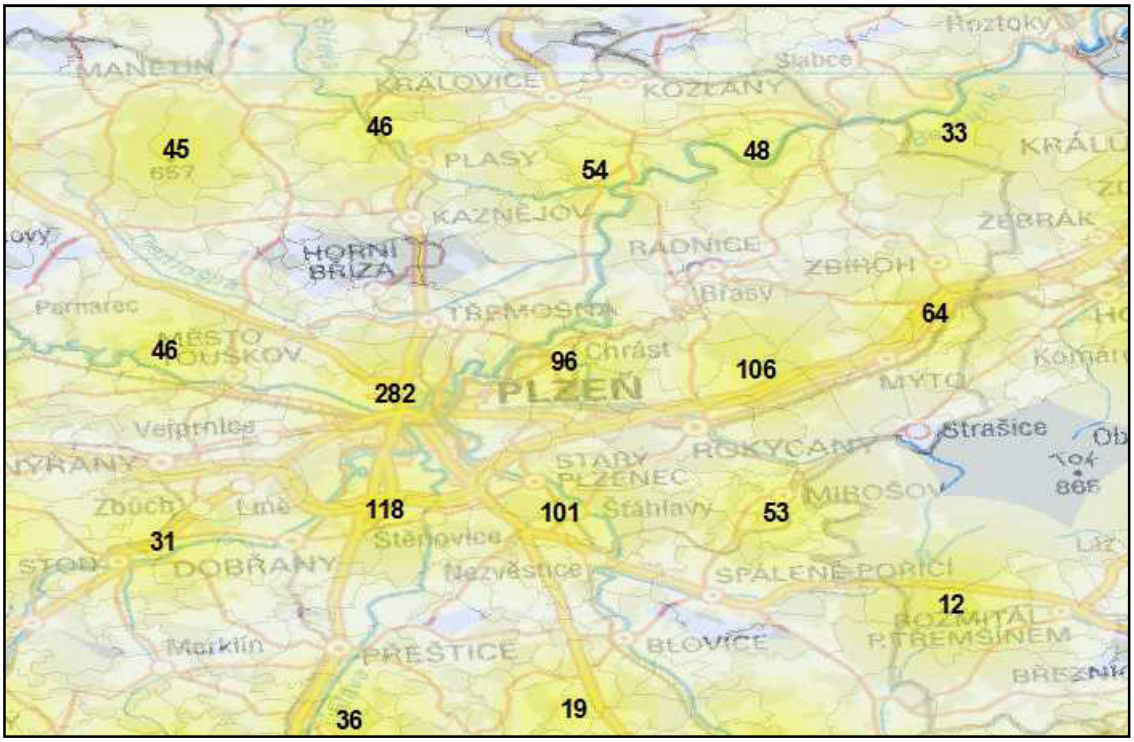
Tab. 7.1: Stupnice pro bodové znaky

Interval	Počet bodů ve shluku	Velikost znaku	Barva znaku
1	1-9	20	velmi světle modrá
2	10-99	28	světle modrá
3	100-999	36	modrá
4	1000 a více	44	tmavě modrá

Taková stupnice není ideální, ale byla zvolena proto, aby v intervalech byla čísla vždy o stejném počtu číslic. To pak umožňuje velikost znaku přizpůsobit tak, aby byl znak co možná nejmenší, ale aby se do něj vešla celá číslice. Znaky pak nejsou zbytečně velké a mapu příliš graficky nezatěžují. Tato reprezentace shluků je velmi jednoduchá, výhodou je také to, že ukazuje přesný počet bodů ve shluku. Nevýhodou je, že neukazuje oblast, ze které se body spojily do jednoho shluku. Ukázka této vizualizace byla vytvořena na shlucích, vytvořených metodou SnapToGrid, vytvořený soubor je nazván *Vizualizace1-bodoveznaky.mxd* a její ukázka je na obrázku 7.1.

7.2 Vizualizace 2

Tato vizualizace reprezentuje shluky pomocí ploch, ze kterých se body spojily do shluků. Tyto plochy jsou pouze přibližné, jsou vytvořeny jako buffery kolem těžišť shluků. Ve středu plochy je číslem uveden počet bodů ve shluku. Tyto plochy jsou vyplněny žlutou barvu, která je světlejší od středu plochy k okrajím. Barva nepřechází plynule, ale skokově v šesti pásech. Lidské oko ale rozezná jen poměrně málo odstínů žluté barvy, proto se barevné přechody jeví poměrně plynule. Mělo by platit, že v místech bez žluté barvy by se neměly nacházet žádné body. To sice není vždy splněno, ale bodů se v těchto oblastech nachází minimum.



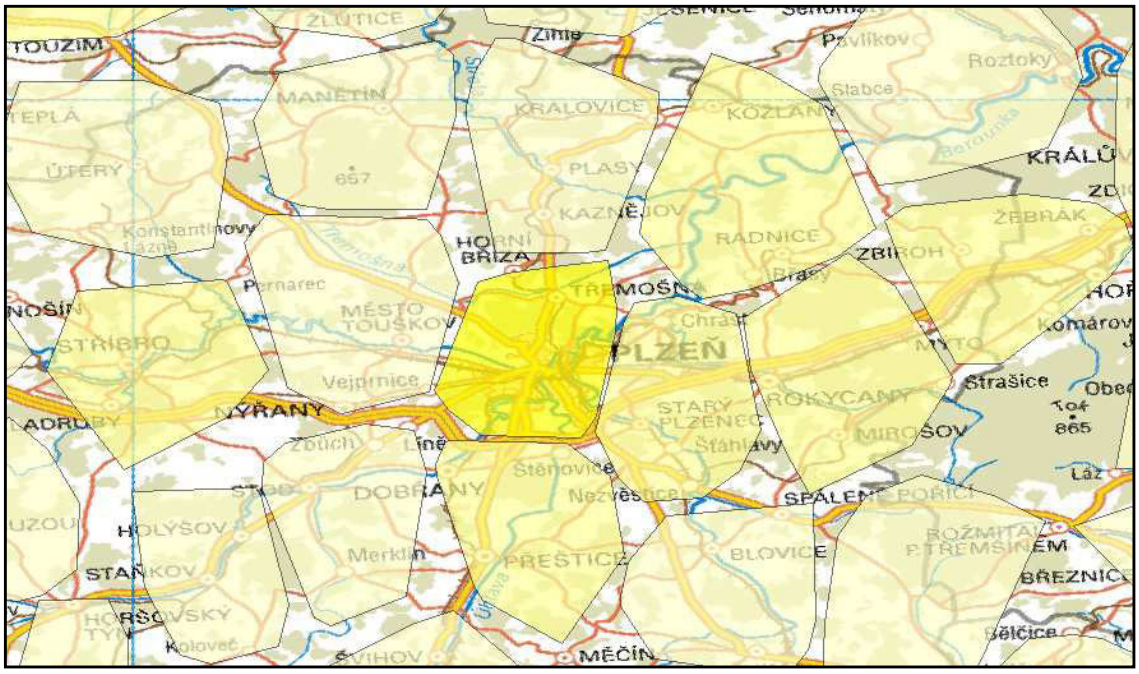
Obr. 7.2: Vizualizace 2 – plošné znaky

Výhodou této vizualizace je to, že znázorňuje přesný počet bodů ve shluku a dává i přibližnou informaci o tom, z jaké oblasti se body spojily do jednoho shluku. Nevýhodou je, že se plochy na mnoha místech slévají, hranice jednotlivých shluků tedy nelze přesně rozeznat. Jisté ale je, že hranice procházejí vždy místy s nejsvětlejší barvou. Tmavší barva ve středu shluků může budit dojem, že v těchto místech je vyšší koncentrace bodů, to ale ve většině případů není pravda. Tento způsob vizualizace byl vytvořen na shlucích metodou SnapToGrid, je uložen pod názvem *Vizualizace2-plosneznaky*.mxd. Ukázka této vizualizace je na obrázku 7.2.

Tato vizualizace byla vytvořena pro měřítko mapy 1:1 000 000 a větší, protože pro menší měřítko nebylo možné zvolit vhodnou velikost bufferu. Docházelo vždy k tomu, že v některých místech docházelo k velkému překrývání ploch a v jiných místech mapy pak zůstávala velká prázdná území.

7.3 Vizualizace 3

Další možností, jak shluky zobrazit plošným způsobem, je vytvořit konvexní obálku bodů, které patří do shluku. Tím vzniknou jakési dlaždice, které přesně znázorňují plochu jednotlivých shluků. Shluky jsou pomocí nepravidelné intervalové stupnice rozděleny do čtyř intervalů metodou přirozených zlomů, podle počtu bodů ve shlucích. Rozdělení touto metodou nejlépe vystihuje specifický charakter dat. Barevná stupnice je volena tak, že shluky s nižším počtem bodů jsou světlejší a naopak shluky s vyšším počtem bodů jsou tmavší. Plochy jsou průhledné, aby bylo možné zobrazit pod nimi mapu. Nevýhodou této reprezentace je to, že nevidíme, kolik bodů obsahují jednotlivé shluky.



Obr. 7.3: Vizualizace 3 – plošné znaky

Tato vizualizace byla vytvořena na shlucích vytvořených metodou k-means s počátečním polem. Je uložena pod názvem *Vizualizace3-plosneznaky.mxd*, její ukázka je na obrázku 7.3.

8 Možnosti práce s časovou složkou dat

Další možnosti práce s testovacími daty nabízí časová složka dat. Jak již bylo zmíněno v kapitole Datový model, časovou složkou dat je datum vzniku bodů. To je uloženo v databázi v tabulce ***cache*** jako datový typ ***date.***

S časovou složkou dat je možné pracovat v databázi, je možné vybírat body z určitého časového intervalu a ty pak generalizovat, v neposlední řadě je možné vytvářet časové mapy, které zobrazují časový vývoj bodů v prostoru a zkoumat možnosti kartografické vizualizace časových změn bodových dat. Problematika vizualizace časových změn prostorových dat je podrobně řešena v diplomové práci [18].

Jako ukázka možnosti práce s časovou složkou testovacích dat byla vytvořena animovaná mapa *casovamapa.avi*. Tato mapa zobrazuje vývoj bodů geocachingu v České republice od roku 2004. Časový interval byl zvolen jeden rok. Generalizace bodů v jednotlivých letech byla provedena metodou k-means. K zobrazení shluků byla použita vizualizace 1, navržená v předchozí kapitole.

Animovaná mapa byla vytvořena v programu ArcGIS. Postup tvorby animované mapy v ArcGIS je podrobně popsán v [18].

9 Závěr

Jak již bylo řečeno v úvodu, cílem práce bylo navrhnout a otestovat řešení kartografické generalizace bodových dat a následné vizualizace generalizovaných dat. Práce je rozdělena na část teoretickou a část praktickou. V teoretické části práce jsou navrženy a podrobně popsány čtyři metody použitelné ke kartografické generalizaci bodových dat. Jedná se o metodu k-means, Geohash, SnapToGrid a MarkerClusterer. Teoretická část dále popisuje metody a faktory kartografické generalizace a metody kartografické vizualizace, vhodné k reprezentaci bodových data a jejich shluků.

V praktické části práce byl vytvořen jednoduchý databázový model pro testovací data, data byla uložena do databáze, kde na ně byly aplikovány navržené generalizační metody. Pomocí jednotlivých metod byly vytvořeny shluky pro měřítkovou řadu 1:1 500 000, 1:1 000 000, 1:500 000, 1:300 000, 1:200 000 a 1:100 000. Výsledky všech čtyř metod byly mezi sebou porovnány. Jako nejvhodnější metoda se jeví metoda k-means. Shluky vytvořené touto metodou vhodně reprezentovaly použitá data a výsledek byl i vhodný pro následnou kartografickou vizualizaci. Při použití této metody je ale potřeba vhodně volit počáteční podmínky. Velice oblíbená je metoda MarkerClusterer, která spolu s generalizací řeší i kartografickou vizualizaci. Tato vizualizace má ale některé nevýhody, které jsou specifikovány v šesté kapitole. Metoda SnapToGrid a metoda Geohash fungují na principu dělení prostoru mřížkou. Shluky vytvořené těmito dvěma metodami nereprezentují data příliš vhodně a výsledky nejsou vhodné pro kartografickou vizualizaci pomocí metody plošných znaků. Dalším záporem metody Geohash je i to, že neumožňuje vytvářet shluky pro libovolné měřítko mapy.

Po vytvoření shluků následovala kartografická vizualizace. Byly navrženy tři způsoby kartografické vizualizace, jedna založená na metodě bodových znaků, druhé dvě vizualizace jsou založené na metodě plošných znaků. Jako ukázka práce s časovou složkou dat byla vytvořena animovaná mapa, která zobrazuje vývoj bodů od roku 2004.

Na práci je možné dále navázat. Další možnosti vidím právě v časové složce dat, která je uložena v databázi a která byla v rámci této práce poměrně potlačena. Ze subjektivního pohledu byla práce zajímavá a přínosná, při práci jsem získala nové znalosti, především z oblasti práce s databází.

Zdroje

[1] ŠUBA, Radan. On-the-fly generalizace nad daty katastru nemovitostí. Plzeň, 2012. Diplomová práce (Ing.). Západočeská univerzita v Plzni, Fakulta aplikovaných věd. Vedoucí práce Karel Janečka.

[2] RABBI, Atta. Clustering and cartographic simplification of point data set. Stockholm 2012. Master of Science Thesis. Royal Institute of Technology (KTH). ISSN 1653-5227.

[3] SKÁLA, Jiří. Algoritmy pro manipulaci s velkými geometrickými daty. Plzeň, 2012. Disertační práce (Ph.D.). Západočeská univerzita v Plzni, Fakulta aplikovaných věd.

[4] VOLDŘICH, Václav. Metody shlukové analýzy pro klasifikaci dokumentů. Plzeň, 2011. Bakalářská práce (Bc.). Západočeská univerzita v Plzni, Fakulta aplikovaných věd. Vedoucí práce Lucie Skorkovská.

[5] STEINBACH, Michael, George KARYPIS a Vipin KUMAR. A Comparison of Document Clustering Techniques.[online]. University of Minnesota. Dostupné z: http://glaros.dtc.umn.edu/gkhome/fetch/papers/docclusterKDDTMW00.pdf.

[6] JAIN, A. K., M. N. Murty and P. J. Flynn. 1999. Data clustering: a review. 1999. Dostupné z: http://doi.acm.org/10.1145/331499.331504

[7] Introduction to PostGIS: Section 14: Spatial Indexing. OpenGeo [online]. [cit. 2013-03-21].
Dostupné z: http://workshops.opengeo.org/postgis-intro/indexing.html.

[8] Big Data Modeling: Visualizing Geohash. [online]. Chicago, 2013, 7.1.2013. [cit. 2013-03-31]. Dostupné z: http://www.bigdatamodeling.org/

[9] Too many markers!. Google developers: Google maps API [online]. 2010, 13.2.2013 [cit. 2013-04-10].
Dostupné z: https://developers.google.com/maps/articles/toomanymarkers?hl=cs.

[10] BAYER, Tomáš. Algoritmy v digitální kartografii. 1. vyd. Praha: Karolinum, 2008. Učební texty Univerzity Karlovy v Praze. ISBN 978-80-246-1499-1.

[11] VOŽENILEK, Vít., Jaromír KAŇOK a kol. Metody tematické kartografie - Vizualizace prostorových jevů. Univerzita Palackého v Olomouci, 2011. ISBN 978-80-244-2790-4.

[12] DENT, Borden D., Jeffrey S. TORGUSON a Thomas W. HODLER. Cartography: Thematic map design. New York: McGraw-Hill, 2009. ISBN 978-0-07-294382-5.

[13] FIEDLER, Jiří. Harmonizace CAD a GIS dat na území Nečtin. Plzeň, 2011. Bakalářská práce (Bc.). Západočeská univerzita v Plzni, Fakulta aplikovaných věd. Vedoucí práce Jan Ježek.

[14] Geocaching: The Official Global GPS Cache Hunt Site. [online]. [cit. 2013-03-4]. Dostupné z: http://www.geocaching.com/.

[15] BURROUGH, Peter A. a Rachael A. McDONNEL. Principles of geographical information systems. 1st ed. repr. Oxford: Oxford University Press, 1998. Spatial information systems and geostatistics. ISBN 0-19-823365-5.

[16] OBE, Regina a Leo HSU. PostGIS in action. London: Manning Publications, 2011. ISBN 978-1-935182-26-9.

[17] Open GIS Consortium, Inc: OpenGIS Simple Features Specification For SQL [online]. [cit. 2013-03-15].
Dostupné z: http://portal.opengeospatial.org/files/?artifact_id=829.

[18] BRAŠNOVÁ, Klára. Kartografické metody pro vizualizaci časových změn prostorových dat. Plzeň, 2012. Diplomová práce (Ing.). Západočeská univerzita v Plzni, Fakulta aplikovaných věd. Vedoucí práce Otakar Čerba.

Přílohy

1. Vytvoření databázových tabulek

```
CREATE TABLE cache
(
  gid character varying(10) NOT NULL,
  the_geom geometry,
  typ smallint,
  vznik date,
  zalozil character varying(60),
  difficulty integer,
  terrain integer,
  CONSTRAINT "identifikator bodu" PRIMARY KEY (gid),
  CONSTRAINT "cash na typ" FOREIGN KEY (typ)
      REFERENCES typ (cislo_typu) MATCH SIMPLE
      ON UPDATE NO ACTION ON DELETE NO ACTION,
  CONSTRAINT "cash na zakladatel" FOREIGN KEY (zalozil)
      REFERENCES zakladatel (jmeno) MATCH SIMPLE
      ON UPDATE NO ACTION ON DELETE NO ACTION
)
WITH (
  OIDS=FALSE
);
ALTER TABLE cache
  OWNER TO student1;
-- Index: "fki_cash na typ"
-- DROP INDEX "fki_cash na typ";
CREATE INDEX "fki_cash na typ"
  ON cache
  USING btree
  (typ);
-- Index: "fki_cash na zakladatel"
-- DROP INDEX "fki_cash na zakladatel";
CREATE INDEX "fki_cash na zakladatel"
  ON cache
  USING btree
  (zalozil COLLATE pg_catalog."default");

CREATE TABLE zakladatel
(
  jmeno character varying(60) NOT NULL,
  pocet_nalezenych smallint,
  clen_od date,
  CONSTRAINT "identifikator zakladatele" PRIMARY KEY (jmeno)
)
WITH (
```

```
  OIDS=FALSE
);
ALTER TABLE zakladatel
  OWNER TO student1;

CREATE TABLE typ
(
  cislo_typu smallint NOT NULL,
  nazev_typu character varying(30),
  CONSTRAINT "identifikace typu" PRIMARY KEY (cislo_typu)
)
WITH (
  OIDS=FALSE
);
ALTER TABLE typ
  OWNER TO student1;
```

2. Import dat do databáze

```
<?php
/**
 * @author Jindra Marvalová, A10B0097P, Jan Strnádek A10B069P
 */
include "dibi.min.php"; // Databáze
// include "simple_html_dom.php"; // PHP parser - no more needed!!
// Own error handler - Debugging purposes!!!
set_error_handler('exceptions_error_handler');
function exceptions_error_handler($severity, $message, $filename, $lineno) {
  if (error_reporting() == 0) {
    return;}
  if (error_reporting() & $severity) {
    throw new ErrorException($message, 0, $severity, $filename, $lineno);}
}
dibi::connect(array(
    'driver'   => 'postgre',
    'host'     => 'mapserver.zcu.cz',
    'port'     => 5432,
    'username' => 'student1',
    'password' => 'student1',
    'database' => 'marvalova'));
dibi::query('TRUNCATE TABLE [cache] CASCADE');
dibi::query('TRUNCATE TABLE [typ] CASCADE');
dibi::query('TRUNCATE TABLE [zakladatel] CASCADE');
$typy = array(
    1 => 'Tradiční', 2 => 'Multikeš', 3 => 'Mystery',4 => 'Letterbox',5 => 'Earth keš',
    6 => 'Webcam keš',7 => 'Event keš');

$typToIndex = array(
```

```
    "Multi-cache" => 2,"Earthcache" => 5, "Traditional Cache" => 1,
    "Letterbox Hybrid" => 4,"Unknown Cache" => 3,"Webcam Cache" => 6,
    "Event Cache" => 7);
foreach ($typy as $key => $value) {
    dibi::insert('typ', array('cislo_typu' => $key, 'nazev_typu' => $value))-
>execute();}

$addr = array('Jihocesky', 'Jihomoravsky', 'Karlovarsky', 'Kraj Vysocina', 'Liberecky',
'Moravskoslezky', 'Olomoucky', 'Pardubicky', 'Plzensky', 'Praha', 'Stredocesky',
'Ustecky', 'Zlinsky');

$types = array(); $autoru = 0; $kesi  = 0; $vynechano = 0;
foreach ($addr as $adresarName) {
$adresar = opendir("Data/".$adresarName."/");
while ($soubor = readdir($adresar)){
    if ($soubor != '.' && $soubor != '..' && substr($soubor, -3) == 'gpx') {
        $file = file_get_contents("Data/".$adresarName."/".$soubor);
        $cacheData = new stdClass();
         preg_match_all("#\<name\>(.*)\<\/name>#", $file, $out);
        $cacheData->id   = (string) $out[1][0];
        // Name
        preg_match_all("#name\>\<!\[CDATA\[(.*)\]\]\>\<\/groundspeak\:name\>#", $file,
$out);
        $cacheData->name = $out[1][0];
        // Author

preg_match_all("#placed_by\>\<!\[CDATA\[(.*)\]\]\>\<\/groundspeak\:placed_by\>#", $file,
$out);
        $cacheData->author = $out[1][0];
        // Lat / Lon
        preg_match_all("#\<wpt lat\=\"(.*)\" lon\=\"(.*)\"\>#", $file, $out);
        $cacheData->latitude = (double) $out[1][0];
        $cacheData->longitude = (double) $out[2][0];
        // Type
        preg_match_all("#groundspeak\:type\>(.*)\<\/groundspeak\:type\>#", $file, $out);
        $cacheData->type = $out[1][0];

        if (!in_array($cacheData->type, $types)) {
            $types [] = $cacheData->type;}
        // Zalozeno
        preg_match_all("#time\>(.*)\<\/time\>#", $file, $out);
        $cacheData->zalozeno = date('Y-m-d H:i:s', strtotime($out[1][1]));
         // DIfficulty
        //<groundspeak:difficulty>
        preg_match_all("#groundspeak\:difficulty>(.*)\<\/groundspeak\:difficulty>#",
$file, $out);
        $cacheData->difficulty = (int) $out[1][0];
        // Terrain
```

```
        preg_match_all("#groundspeak\:terrain>(.*)\<\/groundspeak\:terrain>#", $file,
$out);
        $cacheData->terrain = (int) $out[1][0];
        // Insert?!
        if (array_key_exists($cacheData->type, $typToIndex)) {
            // Existuje author?
            $a = dibi::select('jmeno')->from('zakladatel')->where('[jmeno] = %s',
$cacheData->author)->count();
            if ($a == 0) {
                // Insert
                $authorData = array(
                    'jmeno' => $cacheData->author,
                    'pocet_nalezenych' => rand(1,5000),
                    'clen_od' => date('Y-m-d H:i:s', rand(1167606000, 1354230000))
                );
                dibi::insert('zakladatel', $authorData)->execute();
                $autoru++;}
                        $exist = dibi::select('gid')->from('cache')->where('[gid] = %s',
$cacheData->id)->count();
            if($exist) { continue; }
            // Insert to DB!
            dibi::query("INSERT INTO [cache] ([gid], [the_geom], [typ], [vznik],
[zalozil], [difficulty], [terrain]) VALUES('".$cacheData->id."',
GeomFromText('Point(".$cacheData->longitude." ".$cacheData->latitude.")', 4326),
".$typToIndex[$cacheData->type].", '".$cacheData->zalozeno."',
'".pg_escape_string($cacheData->author)."', ".$cacheData->difficulty.", ".$cacheData-
>terrain.");");
          $kesi++;
        } else {$vynechano++;}
    }
}
closedir($adresar);}?>
```

3. Vytvoření shluků metodou k-means

```
CREATE TABLE kmeans1500 AS (SELECT kmeans, count(*), ST_Centroid(ST_Collect(the_geom))
AS the_geom FROM (SELECT kmeans(ARRAY[ST_X(the_geom), ST_Y(the_geom)], 30) OVER (),
the_geom FROM cache) AS ksub GROUP BY kmeans ORDER BY kmeans);

CREATE TABLE kmeans1000 AS (SELECT kmeans, count(*), ST_Centroid(ST_Collect(the_geom))
AS the_geom FROM (SELECT kmeans(ARRAY[ST_X(the_geom), ST_Y(the_geom)], 50) OVER (),
the_geom FROM cache) AS ksub GROUP BY kmeans ORDER BY kmeans);

CREATE TABLE kmeans500 AS (SELECT kmeans, count(*), ST_Centroid(ST_Collect(the_geom)) AS
the_geom FROM (SELECT kmeans(ARRAY[ST_X(the_geom), ST_Y(the_geom)], 150) OVER (),
the_geom FROM cache) AS ksub GROUP BY kmeans ORDER BY kmeans);
```

```
CREATE TABLE kmeans300 AS (SELECT kmeans, count(*), ST_Centroid(ST_Collect(the_geom)) AS the_geom FROM (SELECT kmeans(ARRAY[ST_X(the_geom), ST_Y(the_geom)], 300) OVER (), the_geom FROM cache) AS ksub GROUP BY kmeans ORDER BY kmeans);

CREATE TABLE kmeans200 AS (SELECT kmeans, count(*), ST_Centroid(ST_Collect(the_geom)) AS the_geom FROM (SELECT kmeans(ARRAY[ST_X(the_geom), ST_Y(the_geom)], 500) OVER (), the_geom FROM cache) AS ksub GROUP BY kmeans ORDER BY kmeans);

CREATE TABLE kmeans100 AS (SELECT kmeans, count(*), ST_Centroid(ST_Collect(the_geom)) AS the_geom FROM (SELECT kmeans(ARRAY[ST_X(the_geom), ST_Y(the_geom)], 1100) OVER (), the_geom FROM cache) AS ksub GROUP BY kmeans ORDER BY kmeans);

CREATE TABLE kmeans1500buf AS (SELECT kmeans, count(*), st_buffer(ST_ConvexHull( ST_Collect(the_geom)),0) AS the_geom FROM (SELECT kmeans(ARRAY[ST_X(the_geom), ST_Y( the_geom)],30) OVER (), the_geom FROM cache) AS ksub GROUP BY kmeans);

CREATE TABLE kmeans1000buf AS (SELECT kmeans, count(*), st_buffer(ST_ConvexHull( ST_Collect(the_geom)),0) AS the_geom FROM (SELECT kmeans(ARRAY[ST_X(the_geom), ST_Y( the_geom)],50) OVER (), the_geom FROM cache) AS ksub GROUP BY kmeans);

CREATE TABLE kmeans500buf AS (SELECT kmeans, count(*), st_buffer(ST_ConvexHull( ST_Collect(the_geom)),0) AS the_geom FROM (SELECT kmeans(ARRAY[ST_X(the_geom), ST_Y( the_geom)],150) OVER (), the_geom FROM cache) AS ksub GROUP BY kmeans);

CREATE TABLE kmeans300buf AS (SELECT kmeans, count(*), st_buffer(ST_ConvexHull( ST_Collect(the_geom)),0) AS the_geom FROM (SELECT kmeans(ARRAY[ST_X(the_geom), ST_Y( the_geom)],300) OVER (), the_geom FROM cache) AS ksub GROUP BY kmeans);

CREATE TABLE kmeans200buf AS (SELECT kmeans, count(*), st_buffer(ST_ConvexHull( ST_Collect(the_geom)),0) AS the_geom FROM (SELECT kmeans(ARRAY[ST_X(the_geom), ST_Y( the_geom)],500) OVER (), the_geom FROM cache) AS ksub GROUP BY kmeans);

CREATE TABLE kmeans100buf AS (SELECT kmeans, count(*), st_buffer(ST_ConvexHull( ST_Collect(the_geom)),0) AS the_geom FROM (SELECT kmeans(ARRAY[ST_X(the_geom), ST_Y( the_geom)],1100) OVER (), the_geom FROM cache) AS ksub GROUP BY kmeans);
```

4. Vytvoření shluků metodou SnapToGrid

```
CREATE TABLE stg1500 AS (SELECT array_agg(gid) AS ids, COUNT( the_geom ) AS count, ( ST_Centroid(ST_Collect( the_geom )) ) AS center FROM cache GROUP BY ST_SnapToGrid( (the_geom), 0.8, 0.8));

CREATE TABLE stg1000 AS (SELECT array_agg(gid) AS ids, COUNT( the_geom ) AS count, ( ST_Centroid(ST_Collect( the_geom )) ) AS center FROM cache GROUP BY ST_SnapToGrid( (the_geom), 0.6, 0.6));
```

```
CREATE TABLE stg500 AS (SELECT array_agg(gid) AS ids, COUNT( the_geom ) AS count, (
ST_Centroid(ST_Collect( the_geom )) ) AS center FROM cache GROUP BY ST_SnapToGrid(
(the_geom), 0.3, 0.3));

CREATE TABLE stg300 AS (SELECT array_agg(gid) AS ids, COUNT( the_geom ) AS count, (
ST_Centroid(ST_Collect( the_geom )) ) AS center FROM cache GROUP BY ST_SnapToGrid(
(the_geom), 0.2, 0.2));

CREATE TABLE stg200 AS (SELECT array_agg(gid) AS ids, COUNT( the_geom ) AS count, (
ST_Centroid(ST_Collect( the_geom )) ) AS center FROM cache GROUP BY ST_SnapToGrid(
(the_geom), 0.15, 0.15));

CREATE TABLE stg100 AS (SELECT array_agg(gid) AS ids, COUNT( the_geom ) AS count, (
ST_Centroid(ST_Collect( the_geom )) ) AS center FROM cache GROUP BY ST_SnapToGrid(
(the_geom), 0.1, 0.1));

CREATE TABLE stg1500 AS (SELECT array_agg(gid) AS ids, COUNT( the_geom ) AS count, (
st_buffer(ST_ConvexHull(ST_Collect(the_geom)),0)) AS center FROM cache GROUP BY
ST_SnapToGrid( (the_geom), 0.8, 0.8));

CREATE TABLE stg1000 AS (SELECT array_agg(gid) AS ids, COUNT( the_geom ) AS count, (
st_buffer(ST_ConvexHull(ST_Collect(the_geom)),0)) AS center FROM cache GROUP BY
ST_SnapToGrid( (the_geom), 0.6, 0.6));

CREATE TABLE stg500 AS (SELECT array_agg(gid) AS ids, COUNT( the_geom ) AS count, (
st_buffer(ST_ConvexHull(ST_Collect(the_geom)),0)) AS center FROM cache GROUP BY
ST_SnapToGrid( (the_geom), 0.3, 0.3));

CREATE TABLE stg300 AS (SELECT array_agg(gid) AS ids, COUNT( the_geom ) AS count, (
st_buffer(ST_ConvexHull(ST_Collect(the_geom)),0)) AS center FROM cache GROUP BY
ST_SnapToGrid( (the_geom), 0.2, 0.2));

CREATE TABLE stg200 AS (SELECT array_agg(gid) AS ids, COUNT( the_geom ) AS count, (
st_buffer(ST_ConvexHull(ST_Collect(the_geom)),0)) AS center FROM cache GROUP BY
ST_SnapToGrid( (the_geom), 0.15, 0.15));

CREATE TABLE stg100 AS (SELECT array_agg(gid) AS ids, COUNT( the_geom ) AS count, (
st_buffer(ST_ConvexHull(ST_Collect(the_geom)),0)) AS center FROM cache GROUP BY
ST_SnapToGrid( (the_geom), 0.1, 0.1));
```

5. Vytvoření shluků metodou geohash

```
Create table GeoHash5 as (select ST_geohash(the_geom, 5), ST_Centroid(ST_Collect(
the_geom)), count(*) the_geom from cache group by ST_geohash(the_geom, 5));

Create table GeoHash4 as (select ST_geohash(the_geom, 5), ST_Centroid(ST_Collect(
the_geom)), count(*) the_geom from cache group by ST_geohash(the_geom, 4));
```

```
Create table GeoHash3 as (select ST_geohash(the_geom, 5), ST_Centroid(ST_Collect(
the_geom)), count(*) the_geom from cache group by ST_geohash(the_geom, 3));

Create table GeoHash5 as (select ST_geohash(the_geom, 5), ST st_buffer((ST_ConvexHull(
ST_Collect(the_geom))), 0), count(*) the_geom from cache group by ST_geohash(the_geom,
5));

Create table GeoHash4 as (select ST_geohash(the_geom, 5), st_buffer((ST_ConvexHull(
ST_Collect(the_geom))), 0)), count(*) the_geom from cache group by ST_geohash(the_geom,
4));

Create table GeoHash3 as (select ST_geohash(the_geom, 5), st_buffer((ST_ConvexHull(
ST_Collect(the_geom))), 0), count(*) the_geom from cache group by ST_geohash(the_geom,
3));
```

6. Vytvoření shluků metodou MarkerClusterer

```
<!DOCTYPE html>
<html>
  <head>
    <title>BP</title>
       <meta charset="utf-8">
    <style>
      html, body, #map_canvas {
        margin: 0;
        height: 100%;
      }
    </style>
    <script
src="https://maps.googleapis.com/maps/api/js?key=AIzaSyChBTj5UNwCCgNo9TmjX4nF1g908O9ccfk
&sensor=false"></script>
    <script src="MarkerClusterer.js"></script>
    <script>
<?php
include "dibi.min.php"; // připojení databáze
dibi::connect(array(
    'driver'   => 'postgre',
    'host'     => 'mapserver.zcu.cz',
    'port'     => 5432,
    'username' => 'student1',
    'password' => 'student1',
    'database' => 'marvalova'
));

// Načtení dat
$a = dibi::query('SELECT ST_X(the_geom) as lon, ST_Y(the_geom) as lat FROM cache');
$array = array();
```

```
foreach($a as $data) {
    $cache = new stdClass();
    $cache->lon = $data['lon'];
    $cache->lat = $data['lat'];
    $array [] = $cache;
}
echo "var caches = ".json_encode($array)."".PHP_EOL;
?>
      function initialize() {
        var mapOptions = {
          zoom: 7,
          center: new google.maps.LatLng(49.7,15),
          mapTypeId: google.maps.MapTypeId.ROADMAP
        };
        map = new google.maps.Map(document.getElementById('map_canvas'), mapOptions);
        var markers = [];
        for (var i = 0; i < caches.length; i++) {
            var marker = new google.maps.Marker({
                position: new google.maps.LatLng(caches[i].lat,caches[i].lon)
            });
            markers.push(marker);
        }
        var cluster = new MarkerClusterer(map, markers);
      }
      google.maps.event.addDomListener(window, 'load', initialize);
 google.maps.event.addListener(marker, 'click', (function(marker, i) {
    return function() {
      infowindow.setContent(marker.position.toString());
      infowindow.open(map, marker);
    }
  })(marker, i));
    </script>
  </head>
  <body>
     <div id="map_canvas"></div>
  </body>
</html>
```