Tvorba nástrojov na optimalizáciu nepravidelných trojuholníkových sietí pre modelovanie georeliéfu

Bc. Martin Kalivoda

Katedra kartografie, geoinformatiky a diaľkového prieskumu Zeme,
Prírodovedecká fakulta, Univerzita Komenského, Mlynská Dolina,
842 15, Bratislava, Slovenská republika
kalivoda.martin@gmail.com

Abstrakt. Predkladaný príspevok prezentuje vyvinutý softwarový nástroj *pg3angles* rozširujúci databázový systém PostgreSQL a PostGIS o funkcie umožňujúce prácu s nepravidelnými trojuholníkovými sieťami – riešenie topológie TIN, analýza TIN (so zameraním na morfometrickú analýzu), optimalizácia TIN pre modelovanie georeliéfu. V ostatnom čase na trhu GIS pribúda mnoho programových riešení, ktoré sú vďaka *open source* prístupu voľne šíriteľné aj modifikovateľné. Jedným z takýchto programových riešení je projekt *PostgreSQL*. Jedná sa o objektovo-relačný databázový systém, ktorý spolu s priestorovým rozšírením *PostGIS* dokáže narábať s geografickými informáciami. *PostGIS* od verzie *2.0*, ktorá sa začala distribuovať *3.4.2012*, podporuje prácu s nepravidelnými trojuholníkovými sieťami. Od verzie *2.1*, ktorá vyšla *17.8.2013*, je možné generovať TIN z podrobného diskrétneho bodového poľa algoritmom Delaunayovej triangulácie. Doteraz však oficiálne pre *PostGIS* nevyšla žiadna knižnica s funkciami, ktoré by vedeli pracovať s TIN, či už riešením vzťahov medzi trojuholníkmi, analýzami alebo optimalizáciou. Tuto vidíme príležitosť vyplniť vzniknutú medzeru prostredníctvom balíka funkcií, ktorý na základe jasne stanoveného konceptu bude vedieť pracovať s TIN. Keďže podpora TIN v rámci *PostGIS* je len v začiatkoch, veríme, že sme medzi prvými, ktorí tvoria toto rozširujúce riešenie a naša práca bude mať obrovský prínos na poli geoinformatiky. Vyvinutú extenziu *pg3angles* sa chystáme vypustiť pod *open source* licenciou *GNU General public license* a ponúknuť ju na využívanie, štúdovanie a zdokonaľovanie študentom, výskumníkom či len zanietencom modelovania georeliéfu pomocou nepravidelných trojuholníkových sietí.

Kľúčové slová: PostgreSQL, PostGIS, extenzia, TIN, optimalizácia

Abstract. Developing the tools for optimizing triangulated irregular networks that model the georelief. This contribution presents the developed software tool called *pg3angles*, which expands database system *PostgreSQL* and *PostGIS* in functions enabling work with triangulated irregular networks – solution of TIN topology, TIN analysis (focusing on morphometric analysis) and TIN optimization for georelief modeling. Recently, the GIS market is being increased by many GIS software solutions, which are freely modifiable and redistributable due to open source approach. One such a software solution is the *PostgreSQL* project. It is the object-relational database management system handling with geographic information thanks to the spatial extension *PostGIS*. *PostGIS* since version *2.0*, distributed since *3/4/2012*, supports working with the triangulated irregular networks. Since version *2.1*, released *17/08/2013*, it is

possible to generate a TIN from the detailed discrete array of points by the algorithm of Delaunay triangulation. However, officially no functions library, dedicated to work with TIN in *PostGIS*, has been released. We see the opportunity to fill this space with our extended package, which will be able to work with TIN on the basis of clearly defined concept. Because of the TIN support is in the beginning within PostGIS, we believe to be among the first ones, who are creating this expanding solution and our work will have great benefit in the field of geoinformatics. The *pg3angles* extension we would release under the open source *GNU General Public License* and we would like to offer it to its usage, study and improvement to students, researchers or just to lovers of modeling georelief using triangulated irregular networks.

Keywords: PostgreSQL, PostGIS, extension, TIN, optimization

1 Úvod

Príspevok vychádza z diplomovej práce s rovnomenným názvom [1]. Vzhľadom na disponibilný priestor sa zameriavame len na charakteristiku vyvinutej extenzie *pg3angles* – stručne popíšeme postavenie, funkcionalitu a inštaláciu, uvedieme pravidlá pre prácu s *pg3angles*, popíšeme koncepciu modifikovaného geometrického typu *TRIANGLE ZM*, nad ktorým je navrhnutá funkcionalita *pg3angles*, zadefinujeme ponúkané funkcie a ukážeme niekoľko príkladov praktického využitia *pg3angles*.

Digitálne modely georeliéfu sú štruktúrami, nad ktorými sú realizované štúdie a analýzy rôznych vedných či technických odborov. Tieto modely aproximujú a rekonštruujú georeliéf prostredí počítača. Vstupom na rekonštrukciu georeliéfu je podrobné diskrétne bodové pole (PDBP), ktoré nesie najmä polohovú (geometrickú) informáciu, ale môže niesť aj rôzne doplnkové informácie o stave georeliéfu. Georeliéf a jeho geometrickú štruktúru možno reprezentovať v prostredí počítača rôznymi formami. Jednou z nich je rastrová forma, ktorá vznikne interpoláciou na základe uzlových bodov so známou výškou. Druhou formou reprezentácie georeliéfu je vektorová forma. V rámci vektorových foriem zohrávajú dôležitú úlohu nepravidelné trojuholníkové siete (TIN), ktoré vznikajú trianguláciou uzlových (vrcholových) bodov.

Výhodou nepravidelných trojuholníkových sietí oproti rastrovej reprezentácii je ich nenáročnosť na pamäťový priestor, informačná robustnosť, ale taktiež aj tvarová flexibilita. Rastrové formy digitálnych modelov georeliéfu sú naopak náročnejšie na pamäťový priestor kvôli maticovej štruktúre, ktorej informatívna hodnota je častokrát redundantná. Je zaujímavé, že napriek uvedeným benefitom nepravidelných trojuholníkových sietí, je tento spôsob vyjadrenia georeliéfu potláčaný rastrovými formami. Možnou príčinou tohto stavu je malá osveta používateľov geografických informačných systémov, ako aj veľakrát absentujúca zložka analytického aparátu jednotlivých GIS narábajúca s TIN.

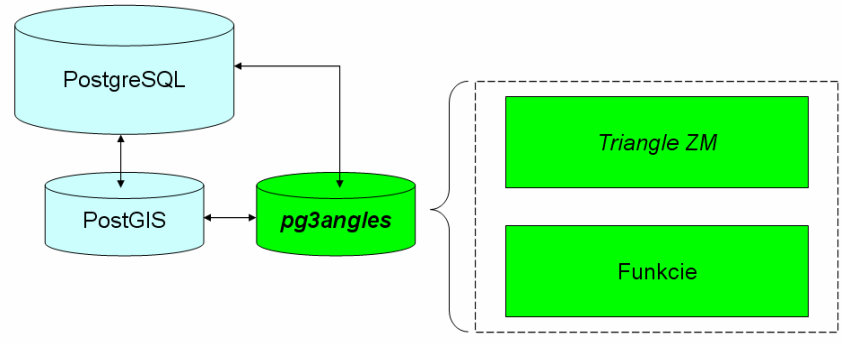
Aj vďaka faktom z predchádzajúceho odseku sme zainteresovaní v modelovaní georeliéfu pomocou TIN, vo vývoji nástrojov pre toto modelovanie a v šírení osvety.

2 Zásuvný modul *pg3angles*

Už z názvu prezentovaného zásuvného modulu možno bádať dve charakterizujúce črty. Prvú črtu evokujú písmená *pg*, ktoré sa v IT svete, najmä v tom databázovom, vzťahujú na všetko, čo súvisí s databázovým systémom PostgreSQL. Druhú črtu naznačuje časť *3angles*, teda slovná hračka vytvorená zo slova *triangles* (z angl. trojuholníky).

Ak sa na zásuvný modul *pg3angles* budeme pozerať z hľadiska jeho postavenia v hierarchii produktov PostgreSQL a PostGIS, všimneme si, že na najvyššom stupni hierarchie je samostatne stojaci databázový systém PostgreSQL bez akýchkoľvek rozšírení. Na druhý stupeň hierarchie umiestňujeme produkty PostGIS a *pg3angles*, ktoré sú závislé na materskom prvku – PostgreSQL, keďže tvoria jeho zásuvné moduly – extenzie. Samotný *pg3angles* je ale závislý okrem materského PostgreSQL aj na rozšírení PostGIS, pretože z neho čerpá priestorové vymoženosti.

Pri bližšom pohľade na *pg3angles* je potrebné všimnúť si, čím konkrétne rozširuje PostgreSQL a PostGIS. Jednak ide o modifikovaný typ geometrie *TRIANGLE ZM*, v rámci ktorého *pg3angles* uchováva informácie o topológii nepravidelnej trojuholníkovej siete, jednak ide o funkcie, ktoré sa všeobecne týkajú práce s nepravidelnou trojuholníkovou sieťou tvorenou trojuholníkmi špeciálneho typu *TRIANGLE ZM*. Postavenie *pg3angles* v hierarchii PostgreSQL a PostGIS spolu s obohacujúcimi komponentmi ilustruje obr. 1.



Obr. 1 Postavenie *pg3angles* v hierarchii PostgreSQL a PostGIS

Funkcionalitu *pg3angles* môžeme stručne zhrnúť do nasledujúcich bodov:

- riešenie topológie TIN,
- morfometrická analýza priestorovej štruktúry reprezentovanej TIN,
- optimalizácia TIN pre modelovanie georeliéfu.

Extenziu *pg3angles* ako súčasť experimentu diplomovej práce [1] uvádzame vo verzii *1.0*. Táto verzia zahŕňa všetky náležitosti uvedené v predchádzajúcom odseku. Na testovacie účely zatiaľ neponúkame zdrojový kód, ale len súbory potrebné na inštaláciu a prácu s *pg3angles* v rámci PostgreSQL a PostGIS. K dispozícii sú na webovej stránke *https://github.com/KaliGIS/pg3angles1.0*.

Pre úspešné zvládnutie inštalácie extenzie *pg3angles 1.0* je potrebné splniť nasledujúce kroky:

0. Stiahnuť potrebné súbory z uvedenej stránky.
1. Súbory *pg3angles--1.0.sql* a *pg3angles.control* nakopírovať do priečinka *SHAREDIR\extension*.
2. Súbor *pg3angles-1.0.dll* nakopírovať do priečinka *LIBDIR*.
3. V rozhraní PostgreSQL pre zadávanie SQL príkazov vykonať príkaz

```
CREATE EXTENSION pg3angles;.
```

Po splnení uvedených krokov je možné plnohodnotne pracovať s *pg3angles*.

Práca s extenziou *pg3angles* je postavená na týchto pravidlách:

- jedna TIN vždy zodpovedá jednej databázovej tabuľke so všeobecným názvom *tintable*, kde jeden riadok predstavuje jeden trojuholník,
- tabuľka *tintable* nutne obsahuje atribúty s názvami *geom* a *tid*, kde *geom* je priestorového dátového typu *geometry* obsahujúci geometriu trojuholníka a *tid* je dátového typu *integer* a predstavuje jedinečný identifikátor každého riadku/trojuholníka,
- trojuholníky v *geom* sú modifikovaného geometrického typu *TRIANGLE ZM* (charakteristika nižšie),
- vrcholové body trojuholníkov TIN sú usporiadané v tabuľke so všeobecným názvom *tintable_points*, ktorá je pomocou funkcií *pg3angles* tvorená v iniciálnej fáze pre konkrétnu TIN,
- tabuľka *tintable_points* nutne obsahuje stĺpce s názvami *pid*, *point* a *triangles*, kde *pid* je dátového typu *integer* a predstavuje jedinečný identifikátor každého riadku/bodu, *point* je priestorového dátového typu *geometry* obsahujúci geometriu bodu a *triangles* je dátového typu *integer[]* obsahujúci pole identifikátorov *tid* trojuholníkov, ktorým jednotlivé (vrcholové) body patria,
- produktom optimalizačnej funkcie *pg3angles* (*TIN_LocalOptimizingIter*) sú tabuľky so všeobecnými názvami *tintable_details_before_iteration_iternum*, *tintable_optimizing_info_iternum* a *tintable_iteration_info*, ktoré slúžia na hodnotenie priebehu optimalizácie. Popis ich stĺpcov je uvedený v charakteristike funkcie *TIN_LocalOptimizingIter* v diplomovej práci [1].

3 Modifikovaný geometrický typ *TRIANGLE ZM*

Pomenovanie *TRIANGLE ZM* predstavuje *WKT* (*well-known text*) kľúč pre určenie druhu priestorového dátového typu – tzv. 4D trojuholníka. *WKT* predstavuje štandard vektorovej reprezentácie dobre čitateľný pre človeka. PostGIS ho má implementovaný.

Štvrtou dimenziou *M* pre účely práce v *pg3angles* definujeme topologické charakteristiky každého jedného prvku typu *TRIANGLE ZM*. Topologické charakteristiky sú tvorené:

- jedinečným identifikátorom *tid* konkrétneho trojuholníka TIN,
- identifikátormi *ntid1*, *ntid2*, *ntid3* susedných trojuholníkov trojuholníka *tid*,
- spresňujúcim identifikátorom *dgt* (charakterizovaný nižšie).

Výhodou kódovania topologických charakteristík do *WKT* reprezentácie je umožnenie zapuzdrenia trojuholníkov trojuholníkovej siete do jedného záznamu geometrického typu *TIN ZM* bez straty informácie o topológii.

WKT kľúč *TRIANGLE*, *TRIANGLE Z*, *TRIANGLE M* alebo *TRIANGLE ZM* vždy pozostáva z troch bodov definujúcich trojuholník (nazvime ho kmeňový). Aby bolo jasné, že ide o plošný element s uzatvorenou hranicou, je k danej trojici bodov priradený štvrtý bod, ktorý musí byť vždy totožný s prvým bodom. Všetky tieto body automaticky preberajú charakteristiku dimenzie *Z*, *M*, *ZM*. Trojuholník typu *TRIANGLE ZM* implicitne tvoria štyri body typu *POINT ZM*, čo umožňuje do hodnôt *M* týchto bodov zakódovať informácie o topológii podľa koncepcie navrhnutej pre *pg3angles*. *M*-hodnota prvého bodu nesie informáciu *ntid1*; o susednom trojuholníku, ktorý leží oproti prvému bodu kmeňového trojuholníka. *M*-hodnota druhého bodu nesie informáciu *ntid2.tid*; o susednom trojuholníku, ktorý leží oproti druhému bodu kmeňového trojuholníka a navyše jedinečný identifikátor kmeňového trojuholníka. *M*-hodnota tretieho bodu nesie informáciu *ntid3.dgt*; o susednom trojuholníku, ktorý leží oproti tretiemu bodu kmeňového trojuholníka. Parameter *dgt* udáva počet cifier identifikátora kmeňového trojuholníka *tid*.

Trojuholník v trojuholníkovej sieti môže mať jeden, dva alebo tri susedné trojuholníky (v mimoriadnych prípadoch aj nula). Identifikátory susedných trojuholníkov *ntid1*, *ntid2*, *ntid3* preberajú hodnotu *tid* jednotlivých susedných trojuholníkov. Existujú aj dve špeciálne hodnoty, ktoré môžu vystupovať v *ntid1*, *ntid2* alebo *ntid3*. Ide o hodnoty *0* a *-1*. Hodnota *0* sa môže vyskytovať len v konfigurácii *ntid1 = ntid2 = ntid3 = 0* a identifikuje stav, v ktorom daný kmeňový trojuholník má síce rozšírenú dimenziu o rozmer *M*, avšak jeho kompletná topologická charakteristika ešte nie je explicitne známa. Tento stav je iniciálnym stavom vzhľadom na účel prác s TIN v *pg3angles*. Nastáva napr. po vygenerovaní trojuholníkovej siete a jej uložení v databáze ešte pred použitím funkcie *TIN_EditNeighboursInfo* upravujúcej jej topologické charakteristiky. Trojuholníky s *0* v *M*-hodnotách nie sú kompatibilné s niektorými funkciami *pg3angles*. Hodnota *-1* v *ntid1*, *ntid2* alebo *ntid3* identifikuje neprítomnosť niektorého alebo viacerých susedných trojuholníkov ku kmeňovému trojuholníku. Vtedy kmeňový trojuholník leží na okraji trojuholníkovej siete.

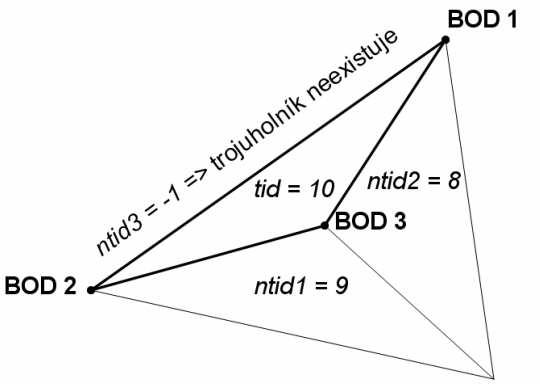
Keďže *M*-hodnoty sú racionálne čísla, bodka vo výraze *ntid2.tid* predstavuje desatinnú bodku. Trojica *dgt* vo výraze *ntid3.dgt* napomáha k jednoznačnej identifikácii konkrétneho *tid*, keďže udáva jeho počet cifier. PostgreSQL podľa štandardu pre prácu s desatinnými číslami v počítači automaticky pri ich vypisovaní ignoruje nuly nasledujúce po poslednej nenulovej číslici za desatinnou čiarkou. Ak *tid = 2*, tak *dgt = 1*. Ak *tid = 20*, tak *dgt = 2*. Ak *tid = 200*, tak *dgt = 3*. Táto koncepcia však umožňuje pracovať najviac s *999 999 999* trojuholníkmi v rámci jednej TIN. Predpokladáme, že prakticky počet trojuholníkov nepresiahne túto hodnotu.

Majme trojuholník vybraný z nepravidelnej trojuholníkovej siete, ktorého *WKT* reprezentácia má tvar

```
TRIANGLE ZM ((10 12 2 9,4.34 8 4.9 8.1,8 9 4.1 -1.2,10 12 2 9)).
```

Jeho topologické charakteristiky sú:

- *tid = 10* (podľa *dgt = 2*),
- *ntid1 = 9*,
- *ntid2 = 8*,
- *ntid3 = -1*.



Obr. 2 Trojuholník *tid = 10* a jeho topologické charakteristiky

4 Funkcie v *pg3angles*

Vo verzii *1.0* ponúka *pg3angles 25* funkcií pre jednoduché, ale aj pokročilejšie operácie nad nepravidelnými trojuholníkovými sieťami. Pre jasnú identifikáciu funkcií z balíka *pg3angles* je v PostgreSQL použitá konvencia predpôn *TIN_*, za ktorými nasleduje výstižný názov funkcie vzhľadom na jej účel.

Disponibilné funkcie možno rozdeliť podľa viacerých hľadísk. Z aspektu programovacieho jazyka použitého na definovanie funkcie je *19* funkcií naprogramovaných v jazyku C a zvyšných *6* funkcií je naprogramovaných v procedurálnom jazyku PostgreSQL (PL/pgSQL). Z hľadiska potreby počtu trojuholníkov ako vstupných parametrov funkcie je *11* funkcií, ktoré operujú nad jedným trojuholníkom bez ohľadu na susedné trojuholníky, *9* funkcií, ktoré vyžadujú viac vstupných trojuholníkov pre úspešné zbehnutie ich algoritmov, *1* funkcia nevyžaduje ako vstup žiaden trojuholník. Ostatné *4* funkcie vyžadujú ako vstup celú TIN reprezentovanú databázovou tabuľkou.

Najdôležitejšie rozdelenie funkcií *pg3angles* je z hľadiska ich zložitosti a hlavného účelu na:

I. Základné funkcie (*20* funkcií):
   1. Topologické funkcie (*5* funkcií),
   2. Analytické funkcie (*12* funkcií),
   3. Funkcie generujúce novú geometriu (*3* funkcie),

II. Kompozitné funkcie (*5* funkcií).

Kompozitné funkcie sú funkcie volajúce vo svojom tele aspoň jednu inú základnú alebo kompozitnú funkciu. Predstavujú sled SQL príkazov za účelom uľahčenia práce užívateľovi.

Vzhľadom na obmedzený priestor neponúkame charakteristiku funkcií *pg3angles* v rámci tohto príspevku. Ich podrobný popis je uvedený v diplomovej práci [1]. Sumarizujeme len prototypy funkcií.

4.1 Prototypy základných funkcií

```
geometry TIN_ExtendM (geometry triangle, double precision
tidentifier);

integer TIN_GetTid (geometry triangle);

geometry TIN_EditNeighboursInfo (geometry[] trianglesarray,
geometry triangle);

integer[] TIN_GetNeighboursTids (geometry triangle);

geometry[] TIN_PntCommonTriangles (geometry triangle, text
tintable, integer vertexnum);

double precision TIN_LinearZ (geometry triangle, geometry
point);

double precision TIN_Area3D (geometry triangle);

double precision[] TIN_PartialDerivativesOfTriangle (geometry
triangle);

double precision TIN_SlopeOfTriangle (geometry triangle);

double precision TIN_AspectOfTriangle (geometry triangle);

double precision TIN_QuasiQuadraticZ (geometry triangle,
geometry[] triangles1, geometry[] triangles2, geometry[]
triangles3, geometry point);

double precision TIN_PartialDerivativesOnVertex (geometry point,
geometry[] triangles);

double precision TIN_SlopeOnVertex (geometry point, geometry[]
triangles);

double precision TIN_AspectOnVertex (geometry point, geometry[]
triangles);

double precision[] TIN_QuasiQuadraticDerivatives (geometry
triangle, geometry[] triangles1, geometry[] triangles2,
geometry[] triangles3, geometry point);

double precision TIN_NormalsAngle (double precision[] normal1,
double precision [] normal2);
```

```
boolean TIN_IsConvexHull (geometry triangle1, geometry
triangle2);

geometry TIN_Centroid (geometry triangle);

geometry TIN_Contours (geometry triangle, double precision
deltaz);

geometry[] TIN_Flip (geometry[] triangles);
```

4.2 Prototypy kompozitných funkcií

```
void TIN_CreateFromPointsDT (text tintablename, text pntstable,
text pntsgeomcol);

void TIN_GrantTinBehaviour (text tintable);

trigger TIN_PrepareForpg3angles (text tintable_points, text
tintable);

void TIN_EditNeighboursInfo (text tintable);

void TIN_LocalOptimizingIter (text tintable, integer iternum);
```

5 *pg3angles* v praxi

Príklad 1: tvorba TIN z PDBP a jej iniciálna úprava pre používanie v *pg3angles*

K dispozícii máme vrstvu (tabuľku) podrobného diskrétneho bodového poľa s názvom *pdbp*, ktorej jeden riadok zodpovedá jednému bodu. Geometria každého bodu je typu *POINT Z* a je uložená v atribúte s názvom *geometria*. Meno vrstvy obsahujúcej trojuholníky bude *tin*. Danú úlohu zrealizujeme zadaním príkazov:

```
SELECT TIN_CreateFromPointsDT ('tin', 'pdbp', 'geometria');
SELECT TIN_EditNeighboursInfo('tin');
```

Výsledkom je tabuľka *tin* naplnená trojuholníkmi typu *TRIANGLE ZM* so známymi topologickými charakteristikami a tabuľka *tin_points* s vrcholovými bodmi trojuholníkov TIN taktiež so známymi topologickými charakteristikami (poľom identifikátorov trojuholníkov *triangles*, ktorým patria).

Príklad 2: výpočet výmery modelovaného územia

K dispozícii máme vrstvu *tin*, ktorá je výsledkom *príkladu 1*. Danú úlohu zrealizujeme zadaním príkazu:

```
SELECT Sum(TIN_Area3D(geom)) AS vymera FROM tin;
```

Výsledkom je jednoriadková tabuľka so stĺpcom *vymera* a kalkulovanou hodnotou. Funkcia *Sum* je tzv. agregačná funkcia štandardne dostupná v PostgreSQL, ktorá počíta sumu výsledkov vstupujúceho príkazu.

Príklad 3: zistenie priemerného sklonu územia
K dispozícii máme vrstvu *tin*, ktorá je výsledkom *príkladu 1*. Danú úlohu zrealizujeme zadaním príkazu:

```
SELECT Sum(TIN_SlopeOfTriangle(geom) * TIN_Area3D(geom)) / Sum(TIN_Area3D(geom)) AS priemerny_sklon FROM tin;
```

Výsledkom je jednoriadková tabuľka so stĺpcom *priemerny_sklon* a kalkulovanou hodnotou.

Príklad 4: vrátenie stranovo-susedných trojuholníkov niektorého trojuholníka
K dispozícii máme vrstvu *tin*, ktorá je výsledkom *príkladu 1*. Chceme získať susedov trojuholníka s identifikátorom *tid = 10*. Danú úlohu zrealizujeme zadaním príkazu:

```
WITH cte AS (SELECT TIN_GetNeighboursTids(geom) AS stids FROM tin WHERE TIN_GetTid(geom) = 10)
SELECT ST_AsText(geom) AS susedia FROM tin, cte
WHERE TIN_GetTid(geom) IN (stids[1], stids[2], stids[3]);
```

Výsledkom je 0-/1-/2-riadková alebo 3-riadková tabuľka so stĺpcom *susedia*, ktorých hodnoty predstavujú *WKT* reprezentáciu susedných trojuholníkov. V príkaze sa objavuje SQL štruktúra WITH s názvom *cte*. Jej názov je ľubovoľný, no v tomto prípade zvolený v zmysle skratky slov *Common Table Expressions*, čo zjednodušene v PostgreSQL predstavuje dočasne vytvorenú tabuľku s totožným názvom *cte*, na ktorú sa dá bežným spôsobom dopytovať. Funkcia *ST_AsText* je funkciou PostGIS, ktorá prevádza binárny tvar geometrie do *WKT* podoby.

Príklad 5: vrátenie trojuholníkov spoločných bodu
K dispozícii máme vrstvy *tin* a *tin_points*, ktoré sú výsledkom *príkladu 1*. Chceme získať všetky trojuholníky, ktoré majú spoločný vrcholový bod. Týmto bodom bude tretí bod trojuholníka s identifikátorom *tid = 10*. Danú úlohu zrealizujeme zadaním príkazu:

```
SELECT ST_AsText(unnest(TIN_PntCommonTriangles(geom, 'tin', 3))) AS spolocne_trojuholniky FROM tin WHERE TIN_GetTid(geom) = 10;
```

Výsledkom je tabuľka so stĺpcom *spolocne_trojuholniky*, kde sú uvedené *WKT* reprezentácie každého jedného trojuholníka spoločného dotknutému bodu. Funkcia *unnest* je štandardnou funkciou PostgreSQL, ktorá prevádza prvky poľa na riadky tabuľky (keďže funkcia *TIN_PntCommonTriangles* vracia pole trojuholníkov).

Príklad 6: odhadovaná výška v ťažisku trojuholníkov
K dispozícii máme vrstvy *tin* a *tin_points*, ktoré sú výsledkom *príkladu 1*. Danú úlohu zrealizujeme zadaním príkazu:

```
SELECT TIN_GetTid(geom) AS tid, TIN_QuasiQuadraticZ(geom,
  TIN_PntCommonTriangles(geom, 'tin', 1),
  TIN_PntCommonTriangles(geom, 'tin', 2),
  TIN_PntCommonTriangles(geom, 'tin', 3),
  TIN_Centroid(geom)) AS vyska_v_tazisku FROM tin;
```

Výsledkom je tabuľka so stĺpcami *tid* a *vyska_v_tazisku* s príslušnými hodnotami pre každý trojuholník z *tin*.

Príklad 7: identifikácia svahov so sklonom nad 5° orientovaných na juh a ich export do novej vrstvy (tabuľky)
K dispozícii máme vrstvu *tin*, ktorá je výsledkom *príkladu 1*. Úloha spočíva vo vytvorení novej tabuľky *juznesvahy*, ktorá bude pozostávať z jedného atribútu typu *geometry* s názvom *geometria*. Následne bude táto tabuľka naplnená výsledkom selekcie trojuholníkov spĺňajúcich zadané kritériá. Danú úlohu zrealizujeme zadaním príkazov:

```
CREATE TABLE juznesvahy (geometria geometry);
INSERT INTO juznesvahy
  SELECT geom FROM tin
WHERE       TIN_SlopeOfTriangle(geom)       >       5       AND
(TIN_AspectOfTriangle(geom) > 315 OR TIN_AspectOfTriangle(geom)
< 45);
```

Príklad 8: vykonanie troch iteračných cyklov optimalizácie TIN
K dispozícii máme vrstvy *tin* a *tin_points*, ktoré sú výsledkom *príkladu 1*. Danú úlohu zrealizujeme zadaním príkazov:

```
SELECT TIN_LocalOptimizingIter('tin', 1);
SELECT TIN_LocalOptimizingIter('tin', 2);
SELECT TIN_LocalOptimizingIter('tin', 3);
```

Výsledkom je optimalizovaná *tin* a sprievodné tabuľky *tin_details_before_iteration_1*, *tin_details_before_iteration_2*, *tin_details_before_iteration_3*, *tin_optimizing_info_1*, *tin_optimizing_info_2*, *tin_optimizing_info_3* a *tintable_iterations_info*. Detailnejší opis algoritmu nájdeme pri charakteristike funkcie *TIN_LocalOptimizingIter* v diplomovej práci [1]. V práci [1] taktiež nájdeme analýzu výsledkov optimalizácie.

6 Záver

Navrhnutý nástroj *pg3angles* si nechceme nechať iba pre seba. Máme v pláne ho vypustiť pod *open source* licenciou *GNU General Public License* umožňujúcou voľné sťahovanie, šírenie a zdokonaľovanie. Na testovacie účely zatiaľ ponúkame súbory potrebné na inštaláciu a prácu s *pg3angles* v rámci systému PostgreSQL a PostGIS pod Windowsom. K dispozícii nie je zdrojový kód. Balík uvedených súborov je dostupný na webovej stránke *https://github.com/KaliGIS/pg3angles1.0*.

Referencie

[1] KALIVODA, M. 2014. Tvorba nástrojov na optimalizáciu nepravidelných trojuholníkových sietí pre modelovanie georeliéfu : diplomová práca. Bratislava : Prírodovedecká fakulta, Univerzita Komenského, 2014. 114 s.