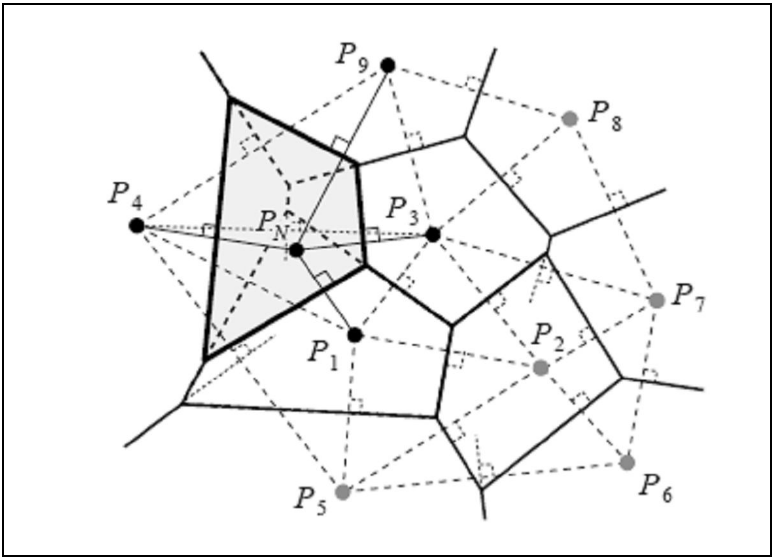
Boldea Costin-Radu

Algoritmică geometrică



EDITURA UNIVERSITARIA
Craiova, 2013

Referenți științifici:
Conf.univ.dr. Boboilă Cristea
Lect.univ.dr. Popîrlan Cristina

Copyright © 2013 Universitaria
Toate drepturile sunt rezervate Editurii Universitaria

Descrierea CIP a Bibliotecii Naționale a României

BOLDEA, COSTIN RADU
Algoritmică geometrică / Boldea Costin-Radu. – Craiova : Universitaria, 2013
Bibliogr.
ISBN 978-606-14-0667-8

514

Apărut: 2013
TIPOGRAFIA UNIVERSITĂȚII DIN CRAIOVA
Str. Brestei, nr. 156A, Craiova, Dolj, România
Tel.: +40 251 598054
Tipărit în România

PREFAȚĂ

Calculatoarele sunt din ce în ce mai mult folosite pentru a rezolva pe scala largă probleme care sunt inerent geometrice. Obiectele geometrice cum ar fi punctele, liniile şi poligoanele formează bazele unei varietăți largi de aplicații importante în teoria algoritmilor.

Astfel se pot identifica mai multe domenii ale informaticii care se ocupă cu rezolvarea problemelor de natură geometrică, cum ar fi reprezentările grafice, procesarea imaginilor, robotica, fabricarea şi proiectarea asistată de calculator, dinamica fluidelor şi „geometria" bazelor de date, pentru a numi câteva dintre ele. Unul dintre scopurile Algoritmicii geometrice este de a oferi instrumentele de bază necesare din care programatorii pot să-şi construiască apoi aplicațiile. S-a făcut un progres semnificativ în acest sens, dar încă e departe de a fi pe deplin realizat.

Algoritmii geometrici sunt importanți de asemenea în studiul modelelor de analiză ale obiectelor fizice, începând de la clădiri şi automobile până la marea categorie a circuitelor integrate. Un proiectant care lucrează cu un obiect fizic are o intuiție geometrica care este greu de implementat în reprezentarea pe calculator a obiectului. Multe alte aplicații implică în mod direct prelucrarea datelor geometrice. De exemplu, o schema politica trucata pentru a diviza o regiune în zone care au populații egale ca număr (si care satisface alt criteriu cum ar fi localizarea majorității membrilor unui partid intr-o singura zona) constituie un algoritm geometric sofisticat. Alte aplicații abunda în matematica şi statistica, unde multe tipuri de probleme care apar pot fi încadrate firesc intr-o reprezentare geometrica.

Cursul îşi propune o introducere succintă în domeniul reprezentărilor grafice geometrice pe calculator, a problematicii şi metodelor sale, precum şi o abordare rapidă a domeniului geometriei computaționale.

Obiectivele cursului.

Cunoaşterea şi înțelegerea noțiunilor fundamentale de reprezentare grafică în C. Introducere în algoritmica geometrică. Prezentarea unor probleme fundamentale ale geometriei computaționale.

Necesar Hardware si software pentru desfasurarea cursului.

- Calculatoare personale cu OS Windows XP sau Windows 7
- Bloodshed Dev-C++ (utilitar gratuit) (http://www.bloodshed.net/devcpp.html) + pachetele grafice (http://www.cs.northwestern.edu/academics/courses/110/html/cs110-glutlib.html)

Capitolul I. Introducere în Algoritmica geometrică

Descriere Generala

Primul capitol este introductiv, propunându-şi familiarizarea cu problematica şi obiectivele algoritmicii geometrice şi a reprezentărilor grafice pe calculator.

Obiective

-Definirea domeniului şi a obiectului de studiu al geometriei computaţionale, ca subdomeniu al algoritmicii

-Familiarizarea cu problematica geometriei computaţionale

1.1 Ce este algoritmica geometrică?

Problemele de geometrie sunt uşor de vizualizat, ceea ce reprezintă dintr-un anumit punct de vedere un handicap. Multe probleme care pot fi rezolvate instantaneu de către o persoană care se uită la o bucată de hârtie (de exemplu: apartenenţa unui punct dat la interiorul unui poligon dat) necesită programe care nu sunt uşoare. Pentru problemele mai complicate, ca şi pentru multe alte aplicaţii, metoda de rezolvare *convenabila* prin intermediul unui calculator poate fi destul de diferită faţă de metoda de rezolvare folosită de către o persoană fizică.

Se presupune în mod uzual că algoritmii geometrici au o istorie lungă datorită naturii constructive a geometriei vechi şi pentru că unele aplicaţii utile sunt atât de larg răspândite, dar de fapt cea mai mare parte din munca depusă în acest domeniu este destul de recentă. Bineînţeles, de multe ori munca vechilor matematicieni este foarte utila în dezvoltarea algoritmilor pentru computerele moderne.

Termenul de **Geometrie computaţională** a fost utilizat prima dată de Marvin Minsky în cartea sa "Perceptroni", care se referea la recunoaşterea formelor (*pattern recognition*) şi de asemenea a fost des folosit la descrierea algoritmilor pentru manipularea curbelor şi suprafeţelor. Totuşi, în mare măsură utilizarea sa curentă este asociată *subdomeniului teoriei algoritmilor* *care studiază proiectarea şi analiza algoritmilor eficienţi pentru probleme derivate din geometrie*. În acest sens am preferat să folosim termenul de **Algoritmică geometrică** pentru acest domeniu, mai potrivit ca descriere decât „geometrie computațională”

Măsurarea calității unui algoritm în geometria computațională se face tradițional prin *timpul asimptotic de rulare al celui mai rău caz*. Astfel, un algoritm care funcționează în timp $O(n)$ este mai bun decât unul care funcționează în timp $O(n \log n)$ care este mai bun decât unul care funcționează în timp $O(n^2)$. (Problema particulară de mai sus poate fi rezolvată în timp $O(n^2 \log n)$ de un algoritm relativ simplu, în $O(n \log n)$ de un algoritm relativ complex şi poate fi aproximat destul de bine de un algoritm mai simplu al cărui timp de rulare este de asemenea $O(n \log n)$). În unele cazuri se ia în considerare *cazul timpului mediu* de funcționare. Totuşi pentru multe tipuri de intrări geometrice este dificil să se definească intrări distributive deoarece ambele sunt uşor de analizat şi reprezentative pentru intrările tipice.

Domeniul geometriei computaționale s-a dezvoltat rapid la sfârşitul anilor ’70, în anii ’80 şi ’90, şi continuă să se dezvolte. Istoric, geometria computaționale s-a dezvoltat ca o generalizare a studiului algoritmilor pentru sortare (*sort*) şi căutare (*searching*) într-un spațiu unidimensional spre probleme care implică intrări multidimensionale. De asemenea, în anumită măsură, s-a dezvoltat ca o limită a teoriei de calcul a graficii, studiind pattern-urile care se nasc normal din proprietățile geometrice.

Datorită istoriei sale, domeniul geometriei computaționale s-a concentrat în mare parte pe probleme în spațiul bidimensional şi s-a extins mai puțin în spațiul tridimensional. Când problemele sunt studiate în spații multidimensionale, de obicei se presupune că dimensiunea spațiului este o constantă mică (să spunem 10 sau mai puțin). Pentru că domeniul a fost dezvoltat de cercetători a căror pregătire era în algoritmi discreți (în opoziție cu analiza numerică) domeniul s-a concentrat deasemenea mai mult pe natura discretă a problemelor de geometrie în opoziție cu problemele continue. *Geometria computațională se ocupă în primul rând cu forme drepte sau plane (linii, segmente de linie, poligoane, planuri şi poliedre) sau forme simple curbate cum sunt cercurile.* Acesta este în contrast, să spunem, cu domenii cum ar fi modelarea în spațiu, care se concentrează pe probleme care studiază curbe şi suprafețe complexe.

1.2 Limitele geometriei computaționale

Există câteva motive relativ normale din cauza cărora geometria computațională nu se va adresa în totalitate amatorilor de geometrie, şi aceste limite ar trebui bine înțelese de la început.

Una dintre acestea este *natura discretă* a geometriei computaționale. Într-un anumit sens orice problemă care este rezolvată pe calculatoare digitale trebuie exprimată în formă discretă, dar multe domenii de aplicație tratează probleme definite pe un spectru mai larg, de la aproximații discrete până la fenomene continue. De exemplu, la procesarea imaginii, imaginea poate fi o discretizare a unei funcții bidimensioanale continue în nuanțe de gri, iar, în robotică, oscilația în sistemele dinamice de control are caracter continuu. Cu toate acestea, există multe aplicații în care obiectele sunt de natură discretă. De exemplu, în sistemele geografice de date, rețelele rutiere sunt discretizate în colecții de segmente de linie.

O altă limită este impusă de faptul că geometria computațională se ocupă în primul rând de *forme drepte sau plane*. Într-un context mai larg, acesta este rezultatul faptului că cercetătorii din geometria computațională nu erau pregătiți în geometrie, ci în programarea algoritmilor discreți. Astfel ei au ales probleme în care geometria şi calculul numeric joacă un rol relativ mic. O mare parte din modelarea în spațiu, dinamica fluidelor şi robotică, se ocupă cu forme care sunt modelate cu suprafețe curbate. Totuşi este posibil să se aproximeze forme curbe cu poligoane plane sau poliedre.

Această presupunere a îngăduit geometriei computaționale să se ocupe de elementele combinatorice ale multora dintre probleme, în opoziție cu preocuparea față de problemele numerice.

Încă o limită este aceea că geometria computațională s-a concentrat în primul rând pe *probleme bidimensionale* şi într-o arie mai restrânsă pe probleme tridimensionale. „Aspectul" mai plăcut al problemelor bidimensionale este că sunt uşor de vizualizat şi uşor de înțeles. Dar multe dintre problemele concrete se plasează contextual în spațiul tridimensional sau spații cu dimensiuni mai mari.

În ciuda acestor limite există încă o arie remarcabilă de probleme interesante cărora geometria computațională li se adresează cu succes. De-a lungul anilor '80 domeniul a dezvoltat multe tehnici pentru proiectarea eficientă a algoritmilor geometrici. Acestea includ bine-cunoscutele metode cum ar fi divide şi vei cucerii şi programarea dinamică, împreună cu câteva metode nou descoperite care par a fi potrivite şi specifice algoritmilor geometrici. O mare parte a acestui curs se va baza pe înțelegerea tehnicii de proiectare eficientă a algoritmilor geometrici.

Între anii 1980 şi începutul anilor 2000 multe dintre problemele deschise ale geometriei computaționale au fost rezolvate în sensul că, teoretic, s-au dezvoltat algoritmi optimi pentru ele. O bună parte dintre aceşti algoritmi au fost greu de implementat din cauza complexității algoritmilor şi structurilor de date pe care le necesitau. Mai mult, aplicațiile care existau erau de multe ori sensibile la erori geometrice care le determinau să producă rezultate greşite sau la blocaje. De exemplu, un programator poate concepe un algoritm pentru determinarea intersecției unei mulțimi de segmente de dreaptă fără a lua în considerație situația când trei segmente de dreaptă se intersectează într-un singur punct. În aceasta situație rară, structura de date folosită poate fi deficientă şi algoritmul se opreşte.

Multe dintre cercetările recente în geometria computațională s-au axat pe încercarea de a face accesibile practicienilor rezultatele teoretice ale algoritmicii geometrice. Acest lucru s-a făcut prin simplificarea algoritmilor existenţi şi producerea de biblioteci întregi de proceduri predefinite.

1.3 Exemple tipice de probleme ale geometriei computaționale

a) **Acoperiri convexe:** Convexitatea este o proprietate geometrică foarte importantă. O mulțime geometrică este *convexă* dacă pentru fiecare două puncte din mulțime, segmentul de dreaptă care le uneşte face parte din mulțime. Una dintre primele probleme identificate în domeniul geometriei computaționale este aceea de a calcula forma convexă cea mai mică, numită *acoperire convexă*, care „acoperă” o mulțime de puncte date.

b) **Intersecții:** Una dintre problemele geometrice de bază este aceea de a determina când două mulțimi de obiecte intersectează o alta. A determina dacă obiectele complexe se intersectează se reduce de cele mai multe ori la a determina care dintre perechile individuale de identități primare (de exemplu: segmente de dreaptă) se intersectează.

c) **Triangularea şi partiționarea:** Triangularea este caracteristică celor mai generale probleme de subdivizare a unui domeniu complex într-o colecție de domenii “simple”, cu largi aplicații în modelare în inginerie şi în geodezie.

d) **Programarea liniară în dimensiuni reduse (2,3):** Multe probleme de optimizare în geometria de calcul pot fi declarate de forma unei probleme de programare liniară, şi anume: să se determine punctele extreme (ex. cele mai ridicate sau cele mai scăzute) care satisfac un set de inegalități liniare. Programarea liniară este o problemă importantă în optimizarea combinatorică, şi de cele mai multe ori oamenii trebuie adesea să rezolve astfel de probleme în spații cu sute sau poate mii de dimensiuni. Totuşi sunt multe cazuri interesante în spații cu număr mic de dimensiuni, dar cu foarte multe constrângeri (exprimate ca inegalități) unde există soluții eficiente foarte simple.

e) **Aranjamente de drepte şi dualitate**: Poate una dintre cele mai importante structuri matematice în geometria computațională este aceea a aranjamentului de drepte (sau la modul general aranjamentul de curbe şi suprafețe). Se dau *n* linii într-un plan, aranjamentul este doar graficul format prin considerarea punctelor de intersecție ca margini şi segmentele de dreaptă care le unesc ca verticale. Se poate arăta că o astfel de structură poate fi construită în timp $O(n^2)$ [1].

f) **Diagramele Voronoi şi triunghiurile Delaunay:** Find dată o mulțime *S* de puncte în spațiu, una dintre cele mai importante probleme este problema celui mai apropiat vecin. Considerând un punct care nu aparține lui *S*, care punct din *S* este cel mai apropiat lui? Una dintre tehnicile folosite pentru rezolvarea acestei probleme este aceea de a subdiviza spațiul în regiuni, potrivit distanțelor la cel mai apropiat punct. Această partiție geometrică a spațiului este numită *diagramă Voronoi*. Structura duală, numită *triungulația Delaunay,* are deasemenea multe proprietăți interesante, fiind cea mai practică triangulare utilizabilă în geodezie şi domeniul măsurătorilor de teren (cadastru), din cauza erorilor mici pe care le introduce.

g) **Regăsirea rapidă a informației (geometrice):** Problemele geometrice de căutare (*searching*) au urmatoarea formă generală: se dă o mulțime de date (de exemplu puncte, drepte, poligoane) care nu se pot modifica, se cere preprocesarea acestei mulțimi de date într-o structură de date astfel încat la un anumit tip de întrebare să se poată răspunde cât mai eficient posibil. De exemplu, o problemă de *căutare* a *celui mai apropiat vecin* este: determinarea punctul din mulțimea de date care este cel mai apropiat de un punct dat. O problemă *de*