Consultes pel veí més proper en arbres k-dimensionals aleatoris

Grau A Q1 Curs 2023-2024

Departament de Ciències de la Computació
Universitat Politècnica de Catalunya

Resum

Aquest projecte té com a objectiu la implementació d'algunes variants dels arbres k-dimensionals aleatoris (*random k-d trees*) així com d'un algoritme que permeti realitzar consultes pel veí més proper en aquests arbres. A continuació haureu d'estudiar experimentalment el cost en el cas mitjà del vostre algoritme i reportar els resultats obtinguts.

El projecte es farà en grups de **4 persones**. Per formalitzar els grups us heu d'apuntar al fitxer compartit `FIB-GRAU-A Equips Projecte Q1 Curs 2023-24` (seguiu l'enllaç). En aquest fitxer trobareu una columna amb tots els vostres noms i una altra columna per a l'ordenació en equips. Moveu els vostres noms de la primera a la segona, i classifiqueu-vos vosaltres com vulgueu **abans del 26 de setembre de 2023**. Els estudiants que no hagin format grup fins aquesta data, seran organitzats en equips pel professorat de l'assignatura.

El lliurament dels materials demanats a aquest projecte es faran en línia via el **Racó FIB** i s'haurà de dur a terme abans de les 23:59 hores del dia **18 d'octubre de 2023**.

En qualsevol moment durant el procés de correcció podríeu ser contactats per part d'algun professor de l'assignatura per tal de resoldre dubtes o fer aclariments sobre el vostre treball.

Totes les comunicacions públiques referents al projecte es duran a terme mitjançant el **Racó FIB** o el **canal de Slack #projecte**.

1 Introducció

Alguna vegada li heu demanat a *Google Maps* per la benzinera més propera? O al *TripAdvisor* per bons restaurants a la zona que envolta la vostra ubicació? Aquestes preguntes són exemples del que formalment anomenem el problema de les cerques associatives (*associative retrieval*), un problema informàtic freqüent a les aplicacions.

Al problema de les cerques associatives es considera una col·lecció F de n registres, on cada registre té una clau que és una k-tupla ($k \geq 2$) $x = (x_0, \ldots, x_{k-1})$ de valors (anomenats atributs o coordenades) extrets d'un domini $D = \Pi_{0 \geq j < k} D_j$, on cada D_j està totalment ordenat.

L'objectiu d'una cerca (o consulta) sobre F és recuperar tots els registres d'F les claus dels quals compleixen amb unes condicions donades. La consulta es considera associativa quan incorpora almenys dos dels atributs de les claus.

Exemples de consultes associatives són: (i) consultes del veí més proper, per recuperar el registre en F amb clau més propera a una clau donada, segons una funció de distància fixada, (ii) consultes de concordança parcial, per a recuperar tots els registres d'F que coincideixen amb els s (de k) atributs especificats, o (iii) consultes per regió, per recuperar tots els registres d'F la clau dels quals es troba dins d'una regió determinada.

Per tractar eficaçment les consultes associatives, l'emmagatzematge dels registres d'F és crucial. Així, les estructures de dades multidimensionals de propòsit general —com els arbres k-dimensionals— són mètodes d'emmagatzematge adequats per donar suport a una àmplia gamma de consultes associatives.

L'elecció correcta de l'estructura de dades que s'adapta millor a les necessitats d'una aplicació requereix una profunda comprensió del seu rendiment davant possibles consultes associatives. Aquest projecte es centrarà precisament a estudiar experimentalment el rendiment d'una estructura de dades de propòsit general per a dades k-dimensionals —els arbres k-dimensionals estàndards— davant d'un tipus de consulta associativa –el veí més proper. O dit d'una altra manera, en dur a terme l'anàlisi experimental del cost en el cas mitjà de consultes pel veí més proper en arbres k-dimensionals.

Per simplificar el que segueix i sense pèrdua de generalitat:

- Identificarem els registres d'F amb les seves claus, que són punts x en un espai k-dimensional amb coordenades $x = (x_0, x_1, \ldots, x_{k-1})$.
- Suposarem que cada x_i pertany a l'interval $D_i = [0, 1]$, i per tant que D és l'hipercub $[0, 1]^k$.

Locality	Longitude	Latitude
(A) Arenys de Mar	2°33′ E	41°33′ N
(B) Barcelona	2°11′ E	41°23′ N
(C) Cardona	1°49′ E	41°56′ N
(D) Delta de l'Ebre	0°45′ E	40°45′ N
(E) Empúries	3°15′ E	42°20′ N
(F) Figueres	2°58′ E	42°14′ N
(G) Girona	2°49′ E	41°59′ N
(H) Hostalric	2°45′ E	41°45′ N
(I) Igualada	1°37′ E	41°35′ N
(J) Jonquera	3°00′ E	42°30′ N
(L) Lleida	0°38′ E	41°37′ N
(M) Manlleu	2°17′ E	42°00′ N
(N) Nùria	2°13′ E	42°30′ N
(O) Olot	2°30′ E	42°11′ N
(P) Puigcerda	1°56′ E	42°26′ N
(Q) Querol	1°30′ E	41°30′ N
(R) Reus	1°06′ E	41°10′ N
(S) Seu d'Urgell	1°28′ E	42°22′ N
(T) Tarragona	1°16′ E	41°07′ N
(U) Ullastret	3°10′ E	42°16′ N
(V) Vic	2°15′ E	41°56′ N
(X) Xert	0°15′ E	40°44′ N

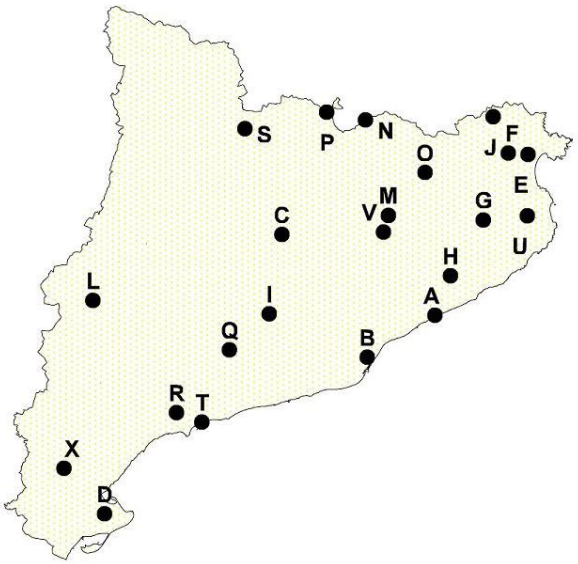

Figura 1: Exemple: Fitxer F de localitats de Catalunya.

1.1 Arbres binaris de cerca k-dimensionals estàndard

En aquest projecte considerarem els arbres binaris de cerca k-dimensionals estàndards, *k-d trees* estàndard per simplificar, definits com segueix.

Definició 1.1 (Bentley75)**.** Un arbre de cerca k-dimensional estàndard T (*k-d tree* estàndard) de mida $n \geq 0$ és una estructura de dades que emmagatzema un conjunt de n registres, on cada un conté una clau consistent en una k-tupla $x = (x_0, \ldots, x_{k-1}) \in D$, on $D = D_0 \times \cdots \times D_{k-1}$, i on cada D_j, $0 \leq j < k$, és un domini totalment ordenat. Sense pèrdua de generalitat, assumirem que $D_j = [0, 1]$ per a tota j, $0 \leq j < k$. L'arbre T és un arbre binari tal que:

- si $n = 0$ és l'arbre buit, o bé,
- si $n > 0$ la seva arrel emmagatzema un registre amb clau x i té associat un discriminant j tal que $j =$ nivell de l'arrel $\mod k$ $(0 \leq j < k)$; els $n-1$ registres restants s'emmagatzemen als subarbres esquerre i dret de T, L i R respectivament, de tal manera que tant L com R són *k-d trees* i es compleix que per a qualsevol clau $u \in L$ $u_j < x_j$ i que per a qualsevol clau $v \in R$ $x_j < v_j$.

Com a exemple considerem que F és el conjunt de localitats de Catalunya que es mostra a la Figura 1 a on cada registre té una clau 2-dimensional corresponent a la longitud i latitud de cada localitat.

A partir d'aquests registres es pot construir, inserint-les claus en ordre alfabètic –per exemple– el *k-d tree* que es mostra a la Figura 2. A la figura, els nodes encerclats en vermell corresponen a nodes on el discriminant és la longitud, mentre que els encerclats en blau tenen com a discriminant la latitud. Es pot apreciar a la figura com tots els nodes del mateix nivell de l'arbre tenen el mateix discriminant i com els discriminants es va assignant cíclicament per nivells: al primer nivell el

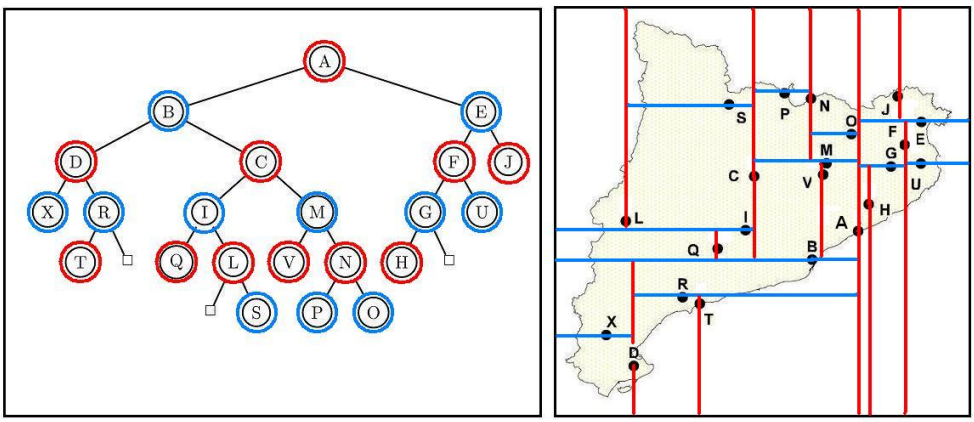

Figura 2: Arbre k-d estàndard per al fitxer de localitats de Catalunya.

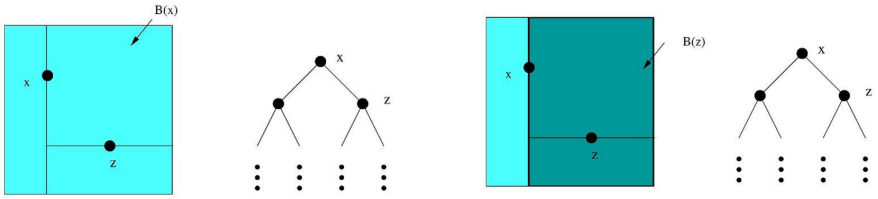

Figura 3: Exemple de *bounding box* en *k-d trees*, on $B(x)$ i $B(z)$ representen els *bounding box* dels nodes x i z, respectivament.

primer discriminant, al segon el segon i així successivament fins s'han esgotat els discriminats i es torna a assignar el primer.

Les següents observacions poden ser útils de cara al projecte:

- Un *k-d tree* de mida n indueix una partició del domini D en $n + 1$ regions, cadascuna corresponent a una fulla del *k-d tree*.
- El *rectangle delimitador* (o en anglès *bounding box*) d'un node amb clau x i discriminant j és la regió de l'espai corresponent a la fulla de l'arbre a la qual s'ha fet la inserció d'x.

  Així, el *bounding box* de l'arrel y amb discriminant 0 de l'arbre sencer és $[0, 1]^K$, el *bounding box* de l'arrel del subarbre esquerre és $[0, 1] \times \cdots \times [0, y_i] \times \cdots \times [0, 1]$, i així successivament, com es pot apreciar a la Figura 3.

1.2 El veí més proper

Com ja hem comentat abans, en aquest projecte ens interessarem en consultes del veí més proper –conegudes en anglès com *nearest neighbour queries*. Definirem aquest tipus de consultes associatives de la manera següent.

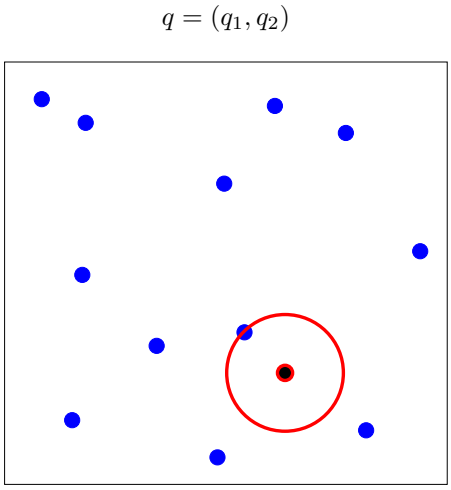

Figura 4: Exemple de *nearest neighbour query* en 2 dimensions fent servir la distància euclidiana. El punt negre és el *query* q i el punt que queda sobre la circumferència vermella és el seu punt més proper, està més a prop de q que qualsevol dels altres punts blaus, els que formen el conjunt F.

Definició 1.2. Donat un conjunt de registres (o punts) k-dimensionals F, un punt k-dimensional q (el *query*) i una funció de distància d definida entre punts k-dimensionals, una consulta pel veí més proper —*nearest neighbor query*— consisteix a trobar el punt $x \in F$ més proper al punt q. És a dir, volem trobar un registre amb clau x tal que:

$$d(q, x) \leq d(q, y), \forall y \in F.$$

A la Figura 4 podeu trobar un exemple gràfic del *nearest neighbor query* del punt q (en negre encerclat de vermell), F és el conjunt dels punts blaus.

Per simplificar el nostre projecte, per les cerques pel veí més proper farem servir la distància euclidiana entre punts k-dimensionals, coneguda també com a norma L_2.

Per la seva dificultat tècnica, l'anàlisi en cas mitjà de les cerques pel veí més proper en arbres binaris k-dimensionals estàndard no s'ha fet formalment (és un problema obert) però existeixen conjectures serioses que diuen que aquest cost és de la forma $\Theta(n^\zeta + \log n)$, amb n el nombre de nodes de l'arbre i on $\zeta \geq 0$ és un valor bastant proper a 0 que depèn de la dimensió k. En aquest projecte volem trobar experimentalment el valor de l'exponent ζ per la distància L_2. El cost de les cerques de veí més proper el mesurarem com el nombre de nodes visitats a l'arbre durant la cerca.

2 Objectius del projecte

Els continguts d'aquest projecte es poden organitzar en tres parts. A la primera part ens encarreguem d'implementar els arbres binaris k-dimensionals estàndards. A la segona part es demana un algorisme per resoldre consultes pel veí més proper en *k-d trees* estàndard. La tercera part es dedicarà a l'anàlisi experimental del cost de cerques pel veí més proper en *k-d trees* estàndard aleatoris.

Part 1: *k-d trees* estàndard

En aquesta part es demana:

(a) Implementar els arbres binaris k-dimensionals estàndard amb tots els procediments necessaris per crear-los buits, inserir-hi claus k-dimensionals, i poder-los destruir un cop hem fet totes les cerques necessàries.

(b) Implementar els procediments necessaris per generar *k-d trees* aleatoris de mida n. Per tal de produir els arbres us proposem que genereu uniformement i independent cadascú dels n punts k-dimensionals a l'interval unitari $[0, 1]^k$ (és a dir, els valors $x_0, x_1, \ldots, x_{k-1}$ de les coordenades de cada punt es generen uniformement i independentment en $[0, 1]$) i inseriu cada punt generat d'aquesta manera al vostre arbre. Hi ha alternatives, però són menys eficients o més complicades.

Part 2: Veí més proper

En aquesta part es demana:

(a) Implementar un algorisme per resoldre el problema de les cerques pel veí més proper en arbres binaris k-dimensionals estàndards aleatoris fent servir la distància euclidiana.

(b) Dotar al vostre algorisme d'un procediment per comptar el nombre de nodes de l'arbre visitats durant la cerca, ja que aquesta serà la manera de mesurar el cost de l'algorisme de cerca.

El vostre algorisme ha d'explotar la desigualtat triangular que satisfà la funció de distància, per tal d'evitar explorar subarbres sempre que sigui possible. Imagineu que a mesura que exploreu l'arbre heu trobat que hi ha un punt a F a distància 0.08 del *query* $q = (0.42, 0.79)$. Suposeu ara que esteu visitant un node x de l'arbre que discrimina per la primera coordenada i que té la clau $(0.65, 0.78)$. El node x no és més proper a q que el més proper trobat fins al moment, és a dir, $d(x, q) > 0.08$ i per tant no hem d'actualitzar de moment el candidat a ser veí més proper. D'altra banda, no hi pot haver cap punt a distància ≤ 0.08 d'q dins del subarbre dret, de manera que el nostre algorisme pot continuar recursivament dins del subarbre esquerre i no explorar en absolut el subarbre dret. Fixeu-vos que si sempre poguéssim descartar un dels dos subarbres i l'arbre fos raonablement equilibrat el cost (mitjà) de la cerca seria $\Theta(\log n)$. Com inserim els punts de F aleatoriament, els *k-d tree* resultants són raonablement equilibrats, però durant una cerca de veí més proper de vegades ens trobarem que s'han d'explorar recursivament els dos subarbres. En qualsevol cas, és inacceptable que el vostre algorisme faci un recorregut complet de l'arbre si es pot evitar, no s'han de fer crides recursives a esquerra i dreta quan podem descartar una de les dues crides. Un algorisme així tindria **sempre** cost lineal respecte a n i no serà acceptat (tot i ser correcte).

Part 3: Experimentació

L'objectiu d'aquesta part del projecte és trobar experimentalment el cost $\Theta(n^\zeta + \log n)$ (amb n el nombre de nodes de l'arbre i $0 \leq \zeta < 1$ un coeficient que depèn de la dimensió k) de les cerques pel veí més proper en arbres binaris k-dimensionals estàndards aleatoris fent servir la distància euclidiana. L'exponent ζ s'ha d'estimar experimentalment.

Podeu procedir de la manera que us sembli millor. Nosaltres us proposem, per exemple, que per a cada possible mida n d'arbres k-dimensionals genereu Q consultes pel veí més proper; en cada consulta la *query* q, serà un punt k-dimensional generat de la mateixa manera que els punts inserits a l'arbre, i s'ha de comptar el nombre de nodes visitats per respondre a cada consulta. Repetiu aquest procediment per a un nombre N d'arbres aleatoris (diferents) i calculeu llavors el corresponent cost mitjà i la seva variància. Repetiu el procediment per diferents valors d'n i per diferents dimensions k (podeu fer que el nombre de dimensions prengui valors entre 2 i 6 compresos). Mostreu els vostres resultats gràficament, evitant gràfiques de més de 2 dimensions.

Heu de justificar els valors escollits de Q i N.

Bonificació

Procediu de la mateixa manera que heu fet per als *k-d trees* estàndard amb altres dues variants d'arbres k-dimensionals: a) els *k-d trees* relaxats, en què els discriminants s'assignen independentment i uniforme als nodes entre les k possibilitats, i b) els *squarish k-d trees*, en què el discriminant assignat a cada node correspon a la coordenada del costat de la *bounding box* més llarg).

Aquest document és intencionadament vague i s'espera que investigueu pel vostre compte totes les tècniques algorísmiques i models que es mencionen aquí. Hi ha molta bibliografia accessible al respecte i no us costarà gens trobar-ne informació.

Informe

Un cop s'ha executat el conjunt complet d'experiments i que es tenen totes les dades reunides, heu de preparar un informe en què:

1. Descriviu breument la vostra implementació d'arbres k-dimensionals i el programa per executar els experiments. Doneu llistes completes del codi com a apèndix del vostre informe.

2. Descriviu breument la configuració experimental, el nombre de combinacions diferents que heu estudiat dels paràmetres i perquè, quantes repeticions i perquè, etc.

3. Proporcioneu taules i gràfics que resumeixin els resultats dels experiments. En particular, heu de donar gràfics que mostrin com el cost mitjà de l'algorisme per resoldre consultes pel veí més proper evoluciona amb n i com varia amb k. Eviteu les gràfiques en 3D.

4. Intenteu obtenir a partir dels resultats experimentals que el cost mitjà de les consultes pel veí més proper en un arbre k-dimensional (estàndard, relaxat o squarish) aleatori de n nodes és $O(n^\zeta + \log n)$. En particular heu d'estimar el valor d'ζ en funció de k i del tipus d'arbre; experimentalment no es pot distingir entre n^ζ si $\zeta \to 0$ i $\log n$, però si les vostres dades experimentals suggereixen que $\zeta = 0$ llavors feu un *fit* de les dades experimentals amb una corba logarísmica.

5. Doneu les vostres conclusions.

Us animem a utilitzar LaTeX per preparar el vostre informe. Per a les parcel·les podeu utilitzar qualsevol dels múltiples paquets que té LaTeX (en en particular, el paquet TikZ+PGF) o utilitzar programari independent com ara gnuplot i després incloure les imatges/gràfics PDF generats així al vostre document.

3 Detalls de l'entrega

El nivell de sofisticació i esforç dedicat a la pràctica és opcional i es tindrà en compte a l'hora d'avaluar-la. Tingueu en compte que la documentació entregada ens ha de permetre valorar el nivell d'assoliment de la competència transversal que hem d'avaluar: **Capacitat d'autoaprenentatge**. En el context del projecte hi ha molts aspectes rellevants relacionats amb aquesta competència: des de l'estudi de noves tècniques i models algorísmics, fins al disseny i anàlisi d'experiments, i la documentació d'aquests tipus de treballs de recerca. La qualificació final del projecte reflectirà la qualitat del vostre aprenentatge, de l'experimentació feta i de la documentació lliurada. La qualitat del codi entregat (programes) es pressuposa i representarà una part molt petita de la qualificació final.

La documentació ha de recollir i presentar la feina feta, les fonts que s'han consultat, el que heu après i els resultats de l'experimentació. En particular és molt important que reflecteixi de forma succinta el que heu après. Si no es compleix aquesta condició, la qualificació final del projecte reflectirà la qualitat de la presentació i no es tindrà en consideració la resta de material lliurat.

Què cal lliurar

Caldrà lliurar una carpeta comprimida (`.zip`) que contingui tots els programes demanats, tots els fitxers addicionals que siguin necessaris per a la compilació i execució de cadascun d'ells així com les instruccions per fer-los anar. A part dels codis dels programes, s'ha d'incloure també l'informe sobre la feina feta. Aquest informe ha de documentar acuradament els algorismes desenvolupats i els experiments realitzats. La qualitat d'aquest informe és un factor molt important per a la qualificació final del projecte.