UNIVERSITÀ DEGLI STUDI DI PADOVA

FACOLTÀ DI INGEGNERIA

Corso di Laurea Magistrale in Ingegneria Informatica

Tesi di Laurea

T.M.C.G.
Tile Manager for Cartographic Generalization

Un approccio al partizionamento di dataset per la generalizzazione automatica

Relatore:
Ch.mo Prof. Sergio Congiu

Laureando:
Pastrello Francesco

A.A. 2011/2012

Contents

Chapter 1

Introduzione

Il presente lavoro di tesi si inserisce all'interno del progetto CARGEN - un progetto di ricerca nato da una convenzione tra la Regione Veneto e il Dipartimento di Ingegneria dell'Informazione dell'Università di Padova - e si pone come obiettivo l'implementazione di un "Tile Manager" per la gestione delle varie fasi del processo automatico per la generalizzazione cartografica effettuato mediante partizionamento del dataset di lavoro.

Una delle problematiche principali nella costruzione di un GIS è la gestione di grandi volumi di dati, che in alcuni casi può diventare critica in quanto comporta la saturazione della memoria del calcolatore sul quale vengono eseguiti processi come l'analisi, il data enrichment e la generalizzazione stessa.

Con questo lavoro, in particolare, si è cercato di implementare una soluzione al processo di generalizzazione, che si adattasse agli algoritmi già presenti all'interno del progetto, e ponesse delle basi per lo sviluppo di futuri algoritmi di derivazione.

Il primo capitolo tratta il tema della cartografia, definendo cosa si intende per processo cartografico, fornendo inoltre alcune informazioni relative al progetto CARGEN, tra cui gli obiettivi e gli strumenti di lavoro.

Nel secondo capitolo viene introdotto il tema del partizionamento, partendo da una definizione generale sino ad arrivare alle problematiche che introduce relativamente al processo cartografico.

Nel terzo capitolo si presenta il T.M.C.G. (Tile Manager for Cartographic Generalization), che porta l'analisi del partizionamento e delle sue criticità proprio al cuore del progetto CARGEN. Dopo aver classificato i vari algoritmi di generalizzazione a seconda delle problematiche che espongono al tema del partizionamento, viene illustrata l'implementazione del TMGC.

Il quarto capitolo fornisce alcuni esempi concreti in cui si è fatto uso del

Tile Manager: si sono portati a termine la "Generalizzazione dei fiumi" e la "Creazione di maglie stradali" mediante il partizionamento del dataset iniziale.

L'ultimo capitolo illustra brevemente la tematica della parallelizzazione, e i motivi per i quali si è scelto di seguire questa strada.

Chapter 2

La cartografia e i GIS

2.1 Cartografia

> *Cartografia [comp. di carta e -grafia; 1858] s.f.: Parte della geografia che si occupa della costruzione di carte geografiche, corografiche e topografiche [25].*

Per comprenderne appieno il significato, è necessario estendere la precedente definizione andando a considerare quella data dall'Associazione Cartografica Internazionale *(ACI, 1996)*, che la interpreta come "il complesso degli studi e delle operazioni scientifiche, artistiche e tecniche che si svolgono a partire dai risultati delle osservazioni dirette o dalla utilizzazione di una documentazione al fine di elaborare ed allestire carte, piante ed altri modi d'espressione atti a risvegliare l'immagine esatta della realtà".

Spesso i dati geografici sono riconosciuti e descritti in modo univoco attraverso una rappresentazione grafica ben definita, attraverso strutture raggruppate o costituite da elementi semplici secondo rappresentazioni tassonomiche di tipo gerarchico. La carta è, quindi, la "rappresentazione in piano dei fenomeni e delle condizioni di fatto della Terra resa in proiezione orizzontale, rimpicciolita, generalizzata e dichiarata nei suoi segni" *(ACI, 1996)*.

Una delle caratteristiche fondamentali di una carta è la scala. La scala della carta è il rapporto numerico delle lunghezze tra la carta ed il terreno (1:10.000 indica che ad 1 mm della carta corrispondono 10.000 mm sul terreno, ossia 10 metri). Il rapporto di riduzione, quindi, si esprime con una frazione che ha per numeratore l'unità e per denominatore il numero per il quale occorre moltiplicare una lunghezza misurata sulla carta per avere la corrispondente lunghezza reale: più piccola è la scala, e minore è la quantità di informazioni che figurano nella carta.

Esistono molteplici modalità di classificazione delle carte, a seconda che si prendano in considerazione diversi aspetti, alcuni dei quali sono "il fattore di scala", "le finalità e i contenuti" e "in base al tipo di proiezione" [21].

Secondo la scala, si usa classificare le carte usando la seguente convenzione:

- planisferi quando rappresentano, senza soluzione di continuità, tutta la superficie terrestre;
- mappamondi quando la rappresentazione è effettuata sopra una sfera;
- carte geografiche o generali quando mostrano una grande estensione della superficie terrestre a piccolissima scala , fino ad 1:2.000.000;
- carte corografiche quando la rappresentazione è a scale comprese tra 1:1.000.000 e 1:200.000;
- carte topografiche quando la rappresentazione è a scale comprese tra 1:100.000 e 1:5.000;
- mappe quando la rappresentazione è a scale comprese tra 1:4.000 e 1:1.000;
- piante per scale grandissime, da 1:500 a valori maggiori;

A queste si aggiungono poi:

- le Carte Tecniche Regionali, che sono rappresentazioni specificamente finalizzate ad interventi sul territorio, in scale da 1:10.000 sino ad 1:2.000.

Considerando i contenuti e le finalità di una carta, invece, è possibile individuare due grandi classi: le carte tematiche e le carte topografiche.

Le carte tematiche sono rappresentazioni di fenomeni o di concezioni astratte qualitative o quantitative limitate ad uno o più temi specifici (ad esempio il piano regolatore di un comune): a loro volta si suddividono in analitiche, complesse, sintetiche, quantitative e qualitative.

Le carte topografiche, invece, vanno a rappresentare una superficie fisica più ampia (scala di riduzione da 10.000 a 200.000) costituita dal terreno con le sue forme, dalle acque (laghi, fiumi, mari), da oggetti concreti e durevoli. Contengono, inoltre, altre indicazioni come confini politici amministrativi, linee o variazioni di colore indicanti l'altimetria e la batimetria e nomi dei luoghi.

2.2 Processo Cartografico

Il processo cartografico, la cui finalità è quella di trasferire sulla carta gli elementi ritenuti significativi per la descrizione della superficie fisica terrestre, è una precisa successione di procedimenti e operazioni, che parte dall'analisi e la definizione delle caratteristiche finali della mappa, passa attraverso l'acquisizione dei dati, e termina con l'effettiva costruzione della mappa e il suo successivo

collaudo.

Durante la fase iniziale, tipicamente, si vanno a definire tutte le caratteristiche tecniche e i contenuti che verranno rappresentati sulla mappa. Per "caratteristiche tecniche" si intendono principalmente la superficie di riferimento adottata, la scala, la proiezione e il tipo di rappresentazione (conforme, equivalente, equidistante). I contenuti della mappa, invece, sono rappresentati da tutte le informazioni che si vogliono collocare nella mappa, e determinano il tipo di astrazione e rappresentazione che saranno utilizzate per trattare i dati in ingresso. Già a questo livello è possibile determinare la natura tematica che la carta andrà ad assumere, ossi focalizzando l'attenzione solo su alcuni particolari aspetti della realtà da rappresentare (cartografia meristica), oppure cercando di avere una visione d'insieme il più completa possibile (cartografia olistica). Già in questa prima fase risulta ben chiara la differenza tra la cartografia cartacea e quella digitale: per la prima, infatti, vi è la produzione della legenda, sostituita nella seconda dalla definizione di un GeoDB.

Terminata la fase di analisi e definizione, si passa alla raccolta dei dati. Due sono le possibili modalità, a seconda che la mappa sia il frutto di una rilevazione o di una derivazione. Nel primo caso la raccolta viene fatta direttamente sul territorio, mediante l'ausilio di sistemi di posizionamento satellitare e fotografie aeree; nel secondo, invece, la mappa viene derivata, utilizzando dati presi da cartografie preesistenti che dovranno avere un grado di dettaglio maggiore (per una corretta astrazione della realtà). Alla fine di questa fase si pone l'assegnazione di un particolare codice per ciascun elemento trovato, a seconda del modello stabilito nella prima fase.

E' questo il momento in cui viene eseguita la generalizzazione cartografica, cioè la fase in cui il cartografo, una volta entrato in possesso di tutte le informazioni necessarie, sceglie e posiziona uno ad uno gli oggetti sulla carta finale, prendendo una serie di decisioni atte a soddisfare le specifiche definite, ma anche a garantire i requisiti classici di ogni mappa (quali la leggibilità e l'usabilità). E', quindi, necessario estrarre dai dati di partenza una loro rappresentazione astratta ma nel contempo efficace e rappresentativa.

Il collaudo è l'ultima fase, in cui si verifica la correttezza e la consistenza della carta. Tra le attività più importanti di collaudo vi è il controllo finale sul terreno mediante operazioni di misura e verifica della rappresentazione cartografica: in questa circostanza si può assistere ad un raffinamento estetico del prodotto, oltre che ad una verifica della validità della rappresentazione creata.

2.3 Gis

L'introduzione dell'informatica nel mondo della cartografia, ha introdotto notevoli cambiamenti nel modo di produrre una mappa, e sicuramente i GIS hanno contribuito notevolmente in questa direzione.

> *Insieme di strumenti software per acquisire, estrarre, elaborare, archiviare e rappresentare dati spaziali del mondo reale [4]*
>
> Insieme di procedure basate sull'uso di sistemi informatici, usate per archiviare ed elaborare dati georeferenziati [2]

I GIS sono sistemi informativi dedicati allo studio e alla gestione di dati geografici, e attraverso questi strumenti è possibile raccogliere, modellare, manipolare, analizzare e presentare dati georeferenziati. L'acquisizione dei dati avviene normalmente scannerizzando le mappe cartacee tradizionali; un altro metodo, invece, si basa sull'utilizzo di file contenenti le coordinate degli elementi geografici, espresse secondo un determinato sistema di riferimento. In entrambi i casi, l'acquisizione è seguita da un processo di astrazione e generalizzazione.

Consentono, perciò, di sovrapporre differenti livelli di informazione relativi ad un'area, ottenendo una migliore comprensione dei fattori che la caratterizzano. Sono, pertanto, uno strumento completo adatto alla rappresentazione del territorio e al trattamento delle informazioni associate agli oggetti georeferenziati.

I dati spaziali - che rappresentano l'informazione territoriale codificata in un sistema informativo geografico - sono tradizionalmente divisi in due classi: raster e vettoriale. Nella prima gli oggetti sono definiti quasi ovunque nel dominio di interesse, e molto spesso sono continui. Sono rappresentati in forma discreta con matrici regolari di attributi (modelli matriciali o raster georeferenziato), reti irregolari di triangoli (modello TIN) oppure con curve di livello. La seconda classe, invece, presenta oggetti discreti e discontinui, delimitati in modo preciso. Sono rappresentati con livelli vettoriali, eventualmente topologici e tabelle di attributi associate: agli oggetti (punti, linee ed aree) sono assegnate le coordinate spaziali congruenti con la base cartografica di riferimento.

In particolare, le primitive vettoriali che costituiscono il modello sono:

1. il punto, definito da una coppia (x,y) o da una terna di coordinate (x, y, z);
2. la linea, costituita da un sequenza ordinata di punti;

3. il poligono, anch'esso costituito da una sequenza ordinata di punti, in cui l'ultimo punto della sequenza corrisponde al primo.

Rispetto alla rappresentazione puramente geometrica degli oggetti ad un GIS viene richiesto di mantenere e gestire le informazioni che riguardano le mutue relazioni spaziali tra i diversi elementi, cioè di strutturare i dati definendone anche la topologia. Oltre all'aspetto geometrico e topologico il GIS deve prevedere anche l'inserimento dei dati descrittivi dei singoli oggetti reali, ovvero degli attributi.

Una volta che i dati vengono memorizzati, è possibile usufruire delle funzionalità di analisi ed elaborazione degli elementi geografici degli attributi fornite dai GIS, quali:

- L'overlay topologico: in cui si effettua una sovrapposizione tra gli elementi dei due temi per creare un nuovo tematismo (ad esempio per sovrapporre il tema dei confini di un parco con i confini dei comuni per determinare le superfici di competenza di ogni amministrazione o la percentuale di area comunale protetta);
- Le query spaziali, ovvero delle interrogazioni di basi di dati a partire da criteri spaziali (vicinanza, inclusione, sovrapposizione etc.)
- Il buffering: da un tema puntuale, lineare o poligonale definire un poligono di rispetto ad una distanza fissa o variabile in funzione degli attributi dell'elemento
- La segmentazione: algoritmi di solito applicati su temi lineari per determinare un punto ad una determinata lunghezza dall'inizio del tema;
- La network analysis: algoritmi che da una rete di elementi lineari (es. rete stradale) determinano i percorsi minimi tra due punti;
- La spatial analysis: algoritmi che utilizzando modelli dati raster effettuano analisi spaziali di varia tipologia, ad es: analisi di visibilità;
- Analisi geostatistiche: algoritmi di analisi della correlazione spaziale di variabili georeferite [17].

Nei GIS le coordinate di un oggetto non sono memorizzate rispetto ad un sistema di riferimento arbitrario o al sistema di coordinate della periferica usata, ma sono memorizzate secondo le coordinate del sistema di riferimento in cui realmente è situato.

La scala di rappresentazione diventa solo un parametro per definire il grado di accuratezza e la risoluzione delle informazioni grafiche. A seconda delle scale elementi piccoli possono non essere visualizzati e si vedono solamente aree di terreno caratterizzate da una stessa quantità [8].

2.4 Generalizzazione

Una delle fasi più importanti del processo cartografico è senza ombra di dubbio la generalizzazione. Qualora si tentasse di rappresentare, ad esempio, una mappa topologica di uno stato, o anche solo di una sua città ad una scala 1:200000, se non venissero portate a termine operazioni di classificazione e semplificazione dell'informazione di partenza avremmo come risultato una mappa illeggibile (un insieme illeggibile di edifici, strade, ec...). Ecco che risulta chiara, perciò, l'importanza che assume questo processo di astrazione della realtà.

L'International Cartographic Association descrive la generalizzazione come "selezione e rappresentazione semplificata dei dettagli che meglio si adatta alla scala e o allo scopo di una mappa" [ICA, 1973]. Tale semplificazione dei dati, quindi, è ben differente dalla semplice riduzione, in quanto viene eseguita in maniera soggettiva seguendo precisi vincoli. Non è quindi un processo puramente scientifico, in quanto tiene in considerazione le abilità e le conoscenze di chi lo esegue. Poiché una mappa è sempre ad una scala minore rispetto alla realtà che rappresenta, l'informazione che andrà a contenere dovrà essere "ristretta" a ciò che risulta essere rappresentabile sulla mappa [12]. Il termine "ristretta" potrebbe far sembrare tale processo passivo: in realtà si tratta più che altro di un processo dinamico di "estrazione" dell'informazione.

Così come per la cartografia, anche la generalizzazione ha subito una notevole spinta innovativa con l'avvento dell'informatica: ecco che la definizione iniziale può essere estesa al concetto spaziale di generalizzazione digitale, intesa come "quel processo responsabile della creazione di visualizzazioni o database geografici ad un minore livello di dettaglio rispetto ai dati di origine, conservando le caratteristiche essenziali dell'informazione geografica sottostante" [20].

Alcuni fattori che devono essere tenuti in considerazione durante il processo di generalizzazione si possono riassumere nei seguenti sei punti [15]:

1. riduzione della complessità - aumentare l'efficacia dell'informazione trasmessa dalla mappa attraverso la diminuzione del numero di oggetti con il passaggio ad una scala inferiore;

2. mantenimento dell'accuratezza spaziale - limitare per quanto possibile l'errore dovuto alla diversa posizione degli oggetti nella mappa rispetto alla realtà, causato dal passaggio ad una scala pià alta;

3. mantenimento dell'accuratezza degli attributi – minimizzare le alterazioni non intenzionali degli attributi della feature;

4. mantenimento della qualità estetica – di non secondaria importanza è la presentazione del risultato finale, che dovrà tener conto di vari fattori quali

"colori utilizzati, simbologia, bilanciamento, layout, ecc.." per garantire la produzione di una mappa esteticamente bella;

5. mantenimento di una logica gerarchica – differenziare gli elementi appartenenti alla stessa categoria ma con importanza differente;

6. applicazione coerente delle regole di generalizzazione – si tratta sostanzialmente di definire chiaramente quali algoritmi eseguire, il loro ordine e di quali parametri di input necessitano.

2.5 Cartografia in Italia

Per quanto riguarda la situazione italiana, il principale ente che si occupa di attività cartografica è l'Istituto Geografico Militare (IGM). Fondato nel 1872, annovera tra le sue prime produzioni la compilazione della "Nuova Carta Topografica d'Italia" (scala 1:100000, la cui realizzazione durò circa trent'anni), a cui fecero seguito le produzioni della "Serie 25V" (scala 1:25000) e, più tardi, della "Serie 50" (scala 1:50000). Ogni serie è in relazione con le altre secondo precisi e definiti rapporti matematici: ciascun foglio di una mappa in scala 1:100.000 è diviso in quattro settori, che rappresentano altrettante mappe della serie 1:50.000 (quadranti); a loro volta ogni quadrante è diviso in quattro parti, dette "tavolette", in scala 1:25.000.

Sin dal 1935 in Italia si avvertiva la necessità di disporre di una cartografia tecnica a grande scala dell'intero territorio nazionale: tra le motivazioni principali, infatti, oltre alla conoscenza dello stesso territorio e delle sue risorse naturali, vi era anche l'eventuale progettazione ed esecuzione dei relativi interventi di sistemazione e sviluppo. Pertanto, la Commissione Geodetica Italiana, rendendosi interprete di questa esigenza, portò avanti studi per la messa a punto delle caratteristiche e dei criteri inerenti alla formazione di una ≪Carta tecnica generale a grande scala dello Stato≫.

Dopo svariati anni si arrivò ad una parziale definizione di tali caratteristiche, istituendo due distinte Sottocommissioni di studio della stessa Commissione Geodetica Italiana (CGI), incaricate rispettivamente di mettere a punto i criteri e le istruzioni utili per la redazione di una guida alle scelte tecniche ed economiche relative alla formazione di carte tecniche alle scale 1:5000 e 1:10000 ed a più grande scala (1:2000 ed 1:1000). L'intento principale era, soprattutto, far sì che gli Enti cartografici dello Stato e le Amministrazioni pubbliche regionali e locali avessero linee guida comuni [7].

A partire dagli anni sessanta, ai sensi della Legge n. 68 del 2 febbraio 1960, l'IGM cominciò ufficialmente a svolgere le funzioni di Ente Cartografico dello Stato assieme ad altri organi, quali l'Istituto idrografico della Marina, la Sezione

fotocartografica dello Stato Maggiore dell'Aeronautica, l'Amministrazione del catasto e dei servizi tecnici erariali, e il Servizio geologico. In particolare, il lavoro dell'IGM era focalizzato sulla stesura della cartografia in piccola scala (1:25000 e superiore), lasciando al Catasto la produzione di carte a grande e media scala (1:10000 e inferiore).

Nel frattempo si fece sempre più urgente la necessità di una cartografia tecnica a grande scala, tanto che si decise per il "Trasferimento alle Regioni a statuto ordinario delle funzioni amministrative statali in materia di urbanistica e di viabilità', acquedotti e lavori pubblici di interesse regionale e dei relativi personali ed uffici" con il D.P.R. 15 gennaio 1972 n. 8. Questa soluzione voleva principalmente soddisfare le più immediate urgenze, con la convinzione che lo sviluppo armonico delle risorse naturali ed umane, la progettazione delle infrastrutture e la gestione del territorio richiedessero una conoscenza molto più approfondita del territorio interessato, rispetto a quella tradizionalmente assicurata dalla Carta topografica d'Italia alla scala 1:25 000 fino a quell'epoca disponibile.

Un ulteriore passo avanti verso la formazione della Cartografia Tecnica Regionale si ebbe con la pubblicazione delle "Norme proposte per la formazione di Carte tecniche alle scale 1:5 000 e 1:10 000" *(CGI, 1973)*, seguite dalla seguente "Guida per le scelte tecniche ed economiche, relativa alla formazione di cartografie generali a più grande scala 1:2000 e 1:1000 *(CGI, 1974)*.

Successivamente, con il DPR 24 luglio 1977 - n. 616, a livello nazionale si stabilì l'ulteriore "Trasferimento funzioni alle regioni in materia di ambiente e territorio", cosicché le regioni poterono cominciare a gestire in maniera autonoma la creazione delle carte regionali. Se da una parte questo fu un motivo di incremento della produzione cartografica e aggiornamento delle mappe, dovuti principalmente alla minor territorio ricoperto, dall'altra portò ad una frammentazione dell'informazione, poiché mancavano degli standard che potessero uniformare tale produzione. Nella sua concreta realizzazione a livello regionale, quindi, la scelta del rapporto di scala della rappresentazione cartografica non risultò sempre uniforme, adeguandosi a criteri operativi ed economici diversificati, in funzione della diversa estensione e situazione del territorio regionale interessato e del suo differente grado di antropizzazione territoriale.

Delle serie cartografiche prodotte dall'IGM alla scala 1:25.000, l'unica ad essere completata è la 25V (vecchio taglio), la meno recente, i cui dati sono mediamente aggiornati al 1960 (ad eccezione di alcuni, il cui aggiornamento risale al 1984). La carta si compone di 3545 tavolette alla scala 1:25 000, le cui dimensioni sono di 7'30" in longitudine e 5' in latitudine. I lavori della successiva Serie 25 - che si sarebbe composta di 2298 elementi, di cui ne sono stati terminati 840, denominati sezioni, che hanno le dimensioni di 6' in latitudine e 10'

in longitudine – furono interrotti dall'Istituto nel 2000 con il passaggio al digitale, rappresentato dalla Serie 25DB. Come la precedente, anche quest'ultima si compone di 2298 sezioni di ugual dimensione, ottenute, però, mediante stereorestituzione numerica o derivate dalla cartografia tecnica regionale numerica.

Anche la cartografia tecnica regionale, grazie allo sviluppo dell'informatica territoriale, registrò i suoi primi progressi, convertendosi gradualmente da un elaborato su supporto cartaceo ad un elaborato numerico su supporto informatico: la Carta Tecnica Regionale Numerica (CTRN). Fu, però, l'avvento della serie 25DB dell'IGM a dare quell'ulteriore spinta innovativa, introducendo il "database geografico" atto a mantenere le informazioni necessarie, concetto che arrivò presto anche agli uffici cartografici regionali. Ciò ebbe un impatto notevole sulla cartografia italiana, aprendo nuove possibilità, tra cui la possibilità di usare la derivazione come mezzo per produrre la nuova serie 25DB: in questo modo sarebbe possibile limitare le rilevazioni dei dati sul territorio, pensando ad un'automatizzazione del processo di derivazione, accelerando notevolmente i tempi di produzione delle sezioni della serie 25DB.

Recentemente la necessità di una più stretta collaborazione tra IGM e regioni si è tradotta nell'istituzione di alcuni gruppi di lavoro, che ha portato alla stesura finale del "Catalogo dei dati territoriali - Specifiche di contenuto per i DB Geotopografici (v1.0)". Questo documento è il risultato dell'attività svolta nell'ambito del Gruppo di Lavoro 2 "Dati geotopografici" istituito dal "Comitato per le regole tecniche sui dati territoriali delle Pubbliche Amministrazioni",

> *"la cui attività ha previsto una sistematica valutazione ed esame di ogni distinto dato/informazione territoriale, organizzati in Strati, Temi e Classi, utilizzando le esperienze professionali dei diversi componenti del Gruppo di Lavoro (Regioni, IGM, IIM, CNIPA, Agenzia del Territorio, Dipartimento della Protezione Civile, Ministero dell'Ambiente, ANCI, ANCITEL, AGEA, UNCEM) e un costante confronto con le attività degli altri Gruppi di Lavoro".*

Da sottolineare come, in questa occasione, si siano seguiti i principi di carattere generale enunciati dalla Direttiva INSPIRE (Direttiva 2007/2/CE del 14 marzo 2007 pubblicata sulla Gazzetta Ufficiale dell'Unione Europea del 25/04/07) che istituisce un'Infrastruttura per l'informazione territoriale nella Comunità europea e l'applicazione ai set di dati territoriali riguardanti i temi elencati negli allegati I, II e III.

Questo documento ha, perciò, definito i due sottoinsiemi del "Catalogo dei Dati Territoriali" per le scale 1:1000/2000 e 1:5000/10000, intesi come contenuti minimi obbligatori per la costituzione di un DB omogeneo a copertura nazionale.

2.6 Progetto Cargen

Il progetto CARGEN - CARtographic GENeralization – è un progetto di ricerca, che dal 2006 vede coinvolti il Dipartimento di Ingegneria Informatica dell'Università di Padova e la Regione Veneto, con la collaborazione dell'Istituto Geografico Militare.

L'obiettivo principale era quello di elaborare nella sua interezza (progettazione, sviluppo e test) un processo automatizzato di generalizzazione cartografica, che partisse dai dati in scala 1:5000 del modello GeoDBR - sviluppato dalla Regione Veneto - per restituire una base di dati coerente con il modello DB25 dell'IGM. In seguito tale obiettivo fu esteso alla generalizzazione di mappe con scala 1:50000. Sebbene i dati di partenza si riferiscono appunto al più recente modello GeoDBR, in questo progetto si è fatto uso anche della CTRN, fornita sempre dalla Regione Veneto. In particolare, la relazione che intercorre tra i modelli ha come punto di partenza la CTRN, che viene utilizzata per popolare il GeoDBR, dal quale si derivano successivamente i dati per popolare il DB25.

Seppur già da tempo, in Europa, la produzione automatizzata di carte a differenti scale attraverso la generalizzazione è prassi ormai consolidata, in Italia la situazione risulta essere diametralmente opposta. Ecco perché questo progetto, essendo tra i primi in Italia, cerchi anche di essere uno stimolo per questo tipo di produzione. Sin dall'inizio si è deciso di intraprendere la strada del raggiungimento di risultati concreti: ad innovativi approcci teorici fine a se stessi si è preferita la realizzazione di un prodotto reale, ossia un software che effettivamente realizzasse la generalizzazione cartografica.

2.6.1 Ambiente di lavoro

Il modello per la gestione dei dati adottato dal progetto CARGEN è di tipo client-server. Il server è costituito da una macchina con installato un DBMS Oracle Spatial 10g, la cui funzione è quella di memorizzare e mantenere i dati spaziali, accessibili tramite query. Inizialmente tutto il carico computazionale era ad appannaggio del server (calcoli geometrici, query spaziali, ecc..), lasciando al client la sola gestione delle geometrie recuperate. Tale architettura è stata via via dismessa, e al fine di garantire migliori prestazioni e un approccio più semplice allo sviluppo di algoritmi (un maggior controllo delle geometrie), si è passati ad una differente modalità di accesso ai dati e successiva visualizzazione (lo strumento utilizzato per quest'ultima è OpenJump).

Grazie al supporto di una libreria sviluppata internamente al Progetto, i dati ora possono essere caricati in RAM e gestiti - lato client - proprio come se fossero delle tabelle. Si è, inoltre, fatto ricorso ad una libreria esterna sviluppata in

Java, la JTS Topology Suite, che fornisce una vasta serie di operatori spaziali, limitando in questo modo la necessità di dover sempre ricorrere al DBMS di Oracle per effettuare interrogazioni spaziali.

2.6.2 JTS Topology Suite

> *La JTS Topology Suite è una libreria open source, scritta interamente in Java, che fornisce una modellazione ad oggetti per le geometrie lineari in uno spazio euclideo bidimensionale [1].*

In particolare, in questa libreria vengono supportate geometrie fondamentali quali Point, LineString e Polygon (rispettivamente geometria puntuale, lineare e areale) e collezioni, con la possibilità di definire una terza dimensione.
La JTS rende disponibili numerose funzioni geometriche, tra le quali citiamo:

1. operatori topologici che realizzano le funzioni di intersezione, differenza, unione;
2. creazione di buffer (sia positivo, che negativo) intorno alle geometrie;
3. costruzione dell'inviluppo convesso di geometrie;
4. costruzione del Minimum Bounding Box,
5. triangolazione
6. funzioni per la semplificazione delle geometrie, come l'algoritmo di Douglas-Peucker;
7. ecc..

Sono presenti inoltre strutture spaziali per l'implementazione di indici spaziali, come il Quad-Tree ed R-Tree, grafi planari ed algoritmi che offrono un modo veloce per la risoluzione di query spaziali.

La versione utilizzata nell'ambito di questa tesi è la JTS 1.11.

2.6.3 OpenJump

OpenJump è una applicazione desktop GIS disponibile sotto la licenza GNU-GPL, tramite la quale è possibile agire su dati locali sia per la visualizzazione che per eventuali interrogazioni e modifiche. Dà la possibilità, inoltre, di agire anche su dati in rete tramite fileserver o tramite DBMS (PostGIS, Oracle, ArcSDE), come pure su dati remoti tramite l'accesso a un server WMS.

E' un GIS derivato da JUMP, sviluppato da Vivid Solution e dalla Refraction Research, Inc., BC Canada: scritto in Java, si basa sulla JTS ed è in grado di

manipolare dati raster o vettoriali, come ad esempio "ESRI-Shapefile". Essendo un progetto opensource, è possibile andare direttamente a modificare il codice sorgente del programma, per adattarlo alle proprie esigenze. Molto interessante, però, risulta essere la sua architettura modulare, che permette di estendere facilmente le funzionalità di base, potendo integrare il proprio codice mediante la realizzazione di plugin.

In questo modo il lavoro di uno sviluppatore viene di molto facilitato: grazie ai plugin, infatti, è possibile visualizzare direttamente il risultato del proprio codice, semplificando e velocizzando notevolmente la fase di test e debug.

In OpenJump i livelli (layer) rappresentano dei "contenitori" di feature, le quali devono appartenere ad un preciso schema dati (FeatureSchema): quest'ultimo specifica il nome e la tipologia degli attributi che costituiscono la feature, simile a quanto accade nelle tabelle dei database. Un layer rappresenta, quindi, una vera e propria tabella, il cui schema dati è specificato dal FeatureSchema. Il layer è interrogabile per mezzo di query, che possono essere sia spaziali che non spaziali. Per migliorare le query spaziali, è possibile associare al layer uno degli indici spaziali forniti dalla JTS.

La versione di OpenJump utilizzata nello svolgimento di questa tesi è la 1.4.0.3.

Chapter 3

Partitioning

3.1 Cenni generali

Da un punto di vista puramente insiemistico, è possibile definire la partizione di un insieme X come una famiglia di sottoinsiemi non vuoti di X Xi — iN, tali che ogni elemento x in X è contenuto in uno e uno solo di questi sottoinsiemi.

In maniera equivalente, un insieme P è una partizione di X sse non contiene l'insieme vuoto e:

1. $\cup$P = X
2. $A \cap B = \emptyset \; if A \in P, B \in PA \neq B$
   dove $\emptyset$ rappresenta l'insieme vuoto.

Gli elementi di P sono chiamati blocchi, parti o celle della partizione [19].

In informatica il problema del partizionamento si viene principalmente a creare poiché, talvolta, si è costretti a lavorare con istanze di grandi dimensioni, che richiedono particolari accorgimenti. Il motivo principale è che, quando la taglia dei dati in input (o prodotti da uno o più algoritmi) supera la taglia della memoria fisica (RAM) della macchina nel quale il problema deve essere risolto, si rischia di andare incontro ad uno stallo, senza riuscire a portare a termine l'esecuzione del/i algoritmo/i.

Una prima soluzione potrebbe essere quella di ignorare il limite dovuto alla capacità della RAM, usufruendo della memoria virtuale per estendere la RAM fisica con l'utilizzo della memoria secondaria, ovvero il disco. Tale soluzione, tuttavia, non risulta sempre praticabile: se da un lato, infatti, si riesce ad evitare il problema di dover gestire in prima persona l'interazione tra RAM e livelli di memoria più capienti (come la memoria secondaria), dall'altro c'è il serio rischio che il sistema operativo non riesca a gestire bene tale comunicazione, andando

a degradare notevolmente le prestazioni (il rischio di stallo, quindi, non sarebbe completamente evitato).

La seconda alternativa, invece, richiede una precisa e consapevole gestione dell'interazione tra memoria secondaria e RAM, avendo ben chiaro che la prima verrà utilizzata per memorizzare i dati, mentre la seconda si occuperà dell'elaborazione degli stessi.

3.2 Il partitioning nel processo cartografico

Uno tra i principali problemi nella costruzione di un GIS è proprio quello della gestione di grandi volumi di dati. Volendo fare una stima, il volume di dati necessario a rappresentare le risorse e l'infrastruttura di una municipalità è dell'ordine di 0,5 – 1 GB per 100000 abitanti, e si possono trovare esempi a supporto di questa tesi che sostengono che una regione urbana necessiterebbe di 14 milioni di punti solo per rappresentare la planimetria della mappa, e questo senza considerare il notevole volume di dati spaziali (coordinate, linee, poligoni, ecc..).

Da considerare, inoltre, vi sono i costi del processo associati all'analisi, al data enrichment e alla generalizzazione, che aumentano considerevolmente con il crescere della zona interessata: è per queste ragioni di performance e di limiti di sistema che si rende necessario sviluppare una tecnica che permetta di dividere lo spazio cartografico in partizioni da elaborare separatamente, possibilmente senza considerare il contenuto informativo proveniente da partizioni vicine (il tutto senza influenzare il risultato finale).

3.2.1 Stato dell'arte

Sul partizionamento di dataset geografici è presente una vasta letteratura, all'interno della quale si possono individuare lavori inerenti al processo di clustering dei dati, [11], [9], [18, a tecniche di partizionamento che fanno uso di grafi [22], [23], quad-tree, R-tree, [14], and GAP tree [1].

Una possibile ed intuitiva visione d'insieme su come partizionare i dati arriva dalla creazione di un dendogramma [2], mentre il Diagramma di Voronoj con il duale possono essere impiegati come metodo per un'efficiente indicizzazione

[1] Il GAP-Tree "Generalized Area Partitioning tree" sviluppato da Van Oosterom *(1991, 1994, 1995)* è un modello di dati spaziali per rappresentare livelli a diverso dettaglio di oggetti all'interno di una partizione planare. Un nodo rappresenta una regione di poligoni generalizzata, che può contenere poligoni di minor importanza, e il cammino foglia-radice corrisponde alla selezione e semplificazione di poligoni durante la fase di generalizzazione

[2] Un diagramma ad albero frequentemente utilizzato per illustrare la disposizione di cluster prodotti da una processo di clustering gerarchico

dell'informazione spaziale.[24].

Altri studi hanno spostato l'attenzione sulla necessità di estrapolare dai dati iniziali le informazioni necessarie per poter ricavare il miglior partizionamento tenendo conto delle esigenze del processo di generalizzazione: alcune soluzioni proposte vanno a combinare differenti categorie di geometrie per creare dei tile "significativi ed intelligenti" [3],[5].

In figura 2.1, tra i vari esempi, si propone come combinare la rete stradale e la linea costiera per ottenere un buon insieme di partizioni chiuse.

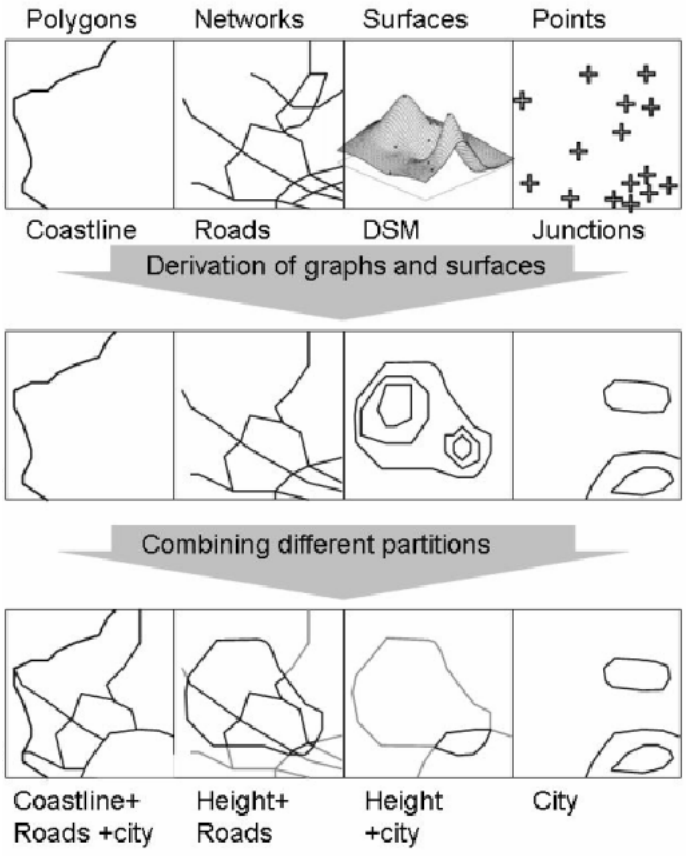

Figure 3.1: Creare più partizioni combinando differenti classi – poligoni, rete stradale, superfici e punti

Altre soluzioni, invece, puntano ad una maggiore consapevolezza del contesto "utile" per la derivazione di una geometria [10]: estendere quindi la par-

tizione analizzata, ampliando il tile con le geometrie vicine, con il solo scopo di utilizzarle in fase di analisi, per portare a termine in maniera precisa e accurata la generalizzazione dei dati interni al tile (solo questi ultimi vengono effettivamente salvati, poiché le geometrie esterne non hanno modo di essere validate).

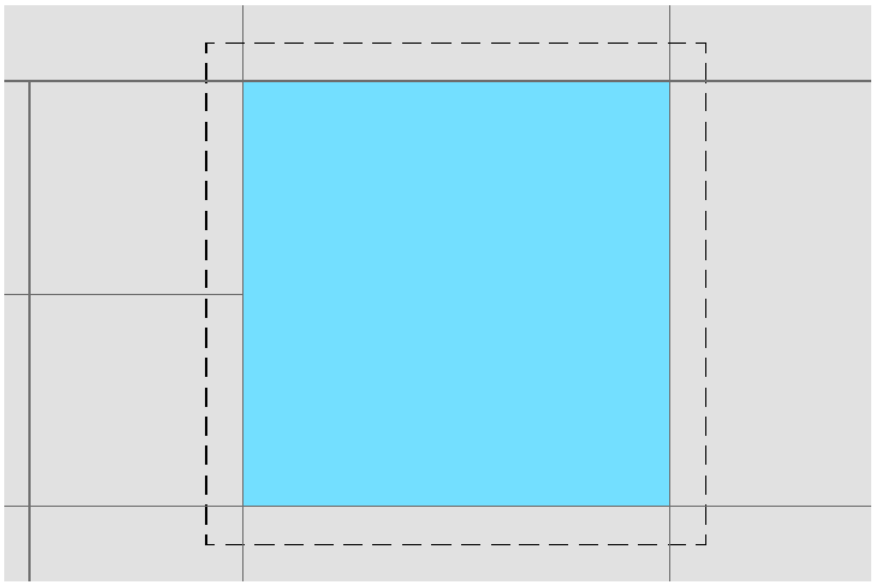

Figure 3.2: Estensione della partizione per tener conto del contesto spaziale.

Il partizionamento suggerito da questo approccio si basa sul quad-tree, che permette di ottenere un miglior controllo del volume di dati su ciascuna partizione, bilanciandone il carico di lavoro. Sfrutta inoltre il concetto di adiacenza: partizioni, quindi, non indipendenti tra loro, ma che si trasmettono a vicenda alcune informazioni aggiuntive ritenute "utili" all'intero processo di generalizzazione. In figura 2.3 osserviamo un'ipotetica esecuzione parallela di sette processi: si inizia con quelle partizioni aventi il maggior numero di partizioni adiacenti non processate, proseguendo man mano fino a quelle con il numero più basso.

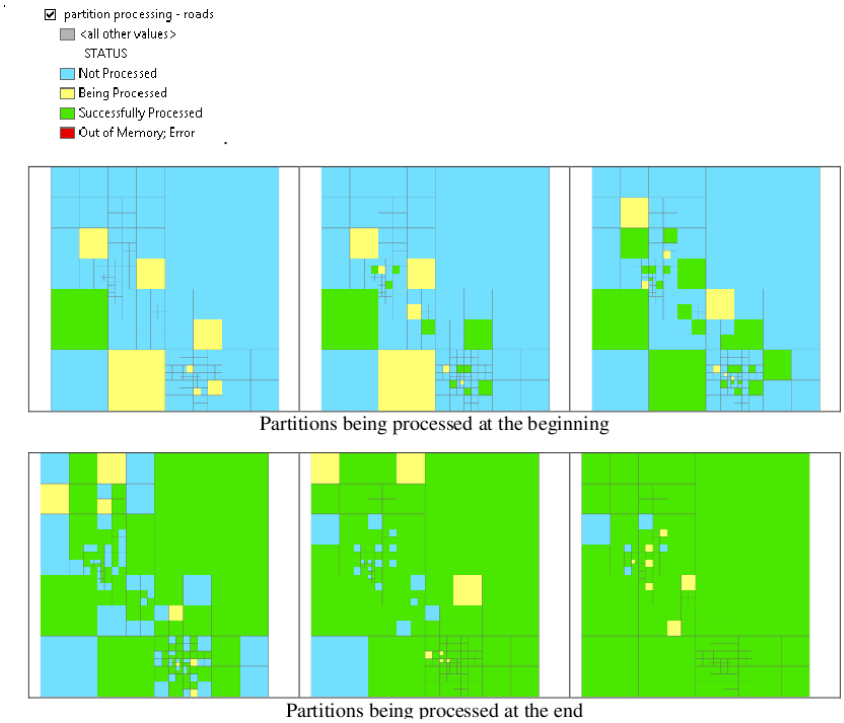

Figure 3.3: Esecuzione parallela di sette processi.

Come si evince dai lavori appena illustrati, le soluzioni al problema del partitioning sono relativamente poche, recenti e parziali, in quanto difficilmente si riescono ad estendere a tutti i tipi di generalizzazione. Le elevate potenze di calcolo dei recenti processori, associate alla grande disponibilità di memoria, garantiscono nella maggior parte dei casi le condizioni ottimali per portare a termine il processo di derivazione.

Negli ultimi anni, però, l'esigenza di lavorare su grandi mappe è diventata sempre più pressante, rendendo necessario lo sviluppo di soluzioni alternative al semplice incremento della potenza di calcolo dei processori.

3.2.2 Problematiche principali

Uno tra gli obiettivi del partitioning è, sicuramente, quello di cercare di bilanciare il carico di lavoro associato a ciascuna partizione: in un sistema con più processori, ad esempio, questo si traduce in una migliore gestione del carico di lavoro per ogni processore.

La difficoltà di questo approccio risiede nella forte dipendenza che sussiste tra il processo di generalizzazione di un oggetto e il "contesto geografico" nel quale l'oggetto stesso viene a trovarsi, poiché va ad influenzare il contenuto di ciascun blocco.

Le criticità principali risiedono proprio sui bordi delle partizioni, in particolare su quelle geometrie che vengono a trovarsi in prossimità di tale limite. E' possibile suddividere la casistica generale in due grandi sottoinsiemi: quello dei casi "continui" e "discreti".

a) Casi continui

I principali esponenti di questo primo insieme sono sicuramente i grafi, in quanto spezzare uno dei loro lati che si trovano sul bordo del tile comporta una conseguente diminuzione di informazione spaziale.

Si pensi, ad esempio, alla **rete idrografica**: tale generalizzazione si basa principalmente sul processo di selezione di un elemento, ossia determinare se un particolare oggetto debba o meno essere eliminato dalla mappa risultante. Tra le principali motivazioni per cui un fiume viene definito "non importante" (e di conseguenza eliminato) vi è la sua lunghezza: lavorando localmente all'interno di una partizione, un tratto di fiume potrebbe essere considerato "corto" poiché non è nota la continuazione all'esterno, e di conseguenza eliminato.

Nell'esempio sottostante è possibile osservare come la naturale ricostruzione del fiume rosso venga ad essere stravolta: il "tratto b" verrebbe ricostruito risalendo sino alla sorgente S2 invece di S1, mentre la parte che interseca il bordo centrale sarebbe eliminata, scollegando di fatto il "tratto b" dal resto del fiume rosso.

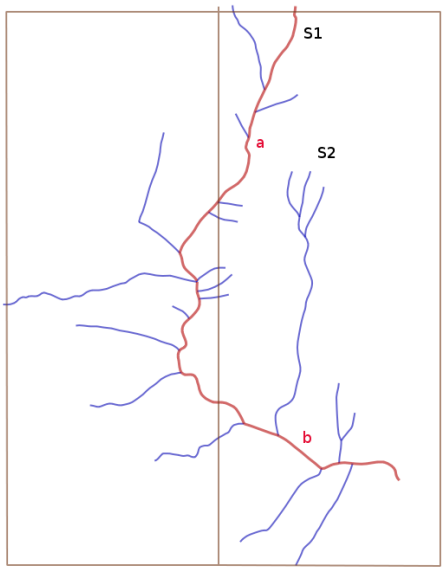

Figure 3.4: Partizionamento tratto fluviale

Così come per la rete fluviale, anche la **rete stradale** presenta problematiche simili, ossia la possibilità che un tratto di strada in prossimità di un bordo, sotto una determinata lunghezza, venga eliminato poiché giudicato "non importante",

senza tener conto di una sua possibile continuazione all'esterno del tile. Tra i possibili esempi è possibile citare quello della generalizzazione di un tratto autostradale, che per essere classificato tale, solitamente, necessita dell'analisi di lunghi tratti.

Anche la gestione degli incroci risulta problematica se non affrontata nella maniera ottimale.

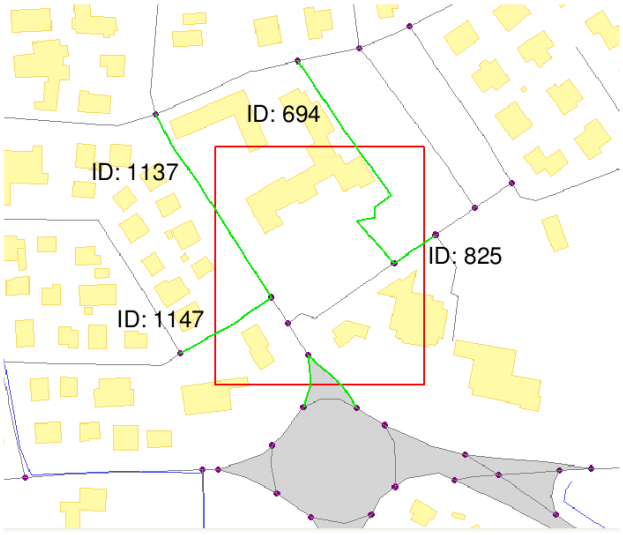

Figure 3.5: Partizionamento rete stradale

Come si evince dalla figura 2.5, l'incrocio in basso (evidenziato in grigio) ha la necessità di una trattazione compatta, che verrebbe meno se i dati all'interno della partizione rossa venissero elaborati e derivati indipendentemente (i due tratti stradali che intersecano il bordo inferiore e che si collegano all'incrocio potrebbero, ad esempio, essere eliminati, portando ad un'errata generalizzazione finale).

I problemi che riguardano i casi continui non fanno riferimento solamente al processo di selezione di una particolare geometria: anche la fase di ricostruzione porta con sé riflessioni aggiuntive. In questa ulteriore classificazione si inseriscono, ad esempio, alcuni algoritmi di semplificazioni di linee che richiedono di specificare un nodo di partenza o ancoraggio [6]. Qualora una di queste geometrie lineari venisse divisa e processata indipendentemente, andrebbe incontro al serio rischio di essere generalizzata erroneamente, a seconda del "punto di rottura" (Figura 2.6a e 2.6b).

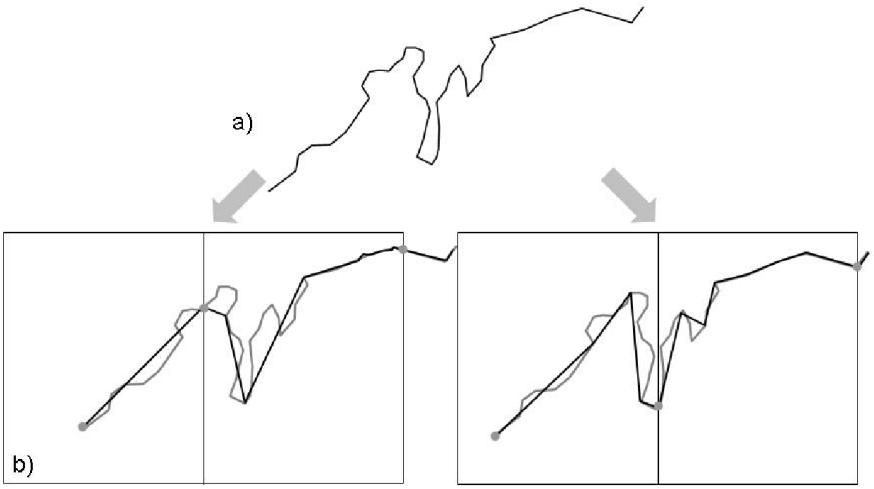

Figure 3.6: Problematica del punto di rottura di geometrie lineari.

b) Casi discreti

Quando si parla di geometrie "discrete" si intendono tutte quelle piccole e grandi geometrie che non fanno parte di un particolare grafo, e per la cui generalizzazione possono risultare necessarie informazioni provenienti dall'ambiente a loro vicino (di geometrie, quindi, non strettamente collegate): si rientra, perciò, nella casistica in cui la derivazione può aver bisogno del contesto spaziale per essere portata a termine correttamente. In figura 2.7 è possibile avere un riscontro visivo immediato: qualora si andassero ad utilizzare partizioni di taglia fissa senza alcun tipo di controllo, si rischierebbe di produrre tre geometrie in output (figura 2.7a) invece delle due che ci si aspetterebbe di avere (figura 2.7b).

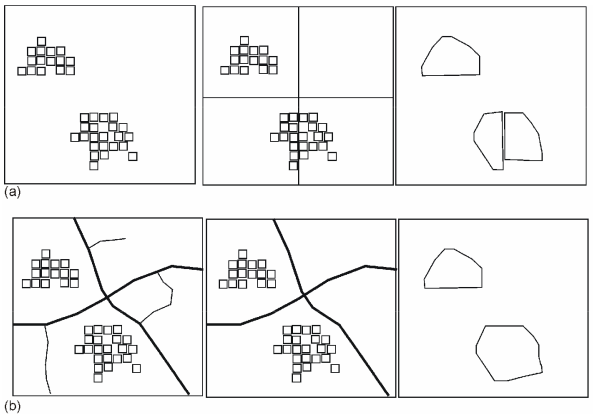

Figure 3.7: Problematica del partizionamento di blocchi di edifici

Spezzare una geometria (o un insieme di geometrie) senza un'opportuna gestione comporterebbe la creazione di nuove identità indipendenti, ognuna

derivata successivamente senza tener conto dell'altra. (figura 2.7).

Tale problematica si riferisce, in particolar modo, alla tipologia di relazioni gerarchiche che possono instaurarsi tra le geometrie a differenti livelli di analisi [16].

In un primo caso si parla di "*dipendenza dall'essere parte di un gruppo*", in quanto si rende necessaria l'identificazione di gruppi significativi per portare a termine operazioni di generalizzazione che non si riuscirebbe ad eseguire considerando gli oggetti indipendentemente, come ad esempio unione e tipificazione.

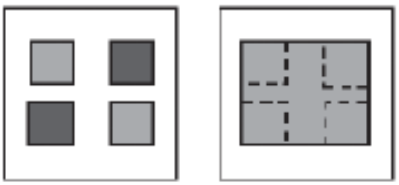

Figure 3.8: Union: fonde in un unico elemento oggetti che precedentemente erano separati o distinti. La scelta degli oggetti da sottoporre a questo tipo di operazione si casa su considerazioni di tipo semantico.

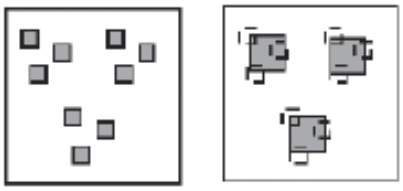

Figure 3.9: Typification: riduce la complessità di un gruppo di oggetti attraverso la loro eliminazione, riposizionamento, allargamento o aggregazione mantenendo la disposizione tipica di quegli oggetti.

Inoltre, l'essere parte di un gruppo significativo può influenzare il modo di generalizzare i suoi elementi (ad esempio, per garantire un certo livello di omogeneità della trasformazione finale, un edificio raggruppato assieme ad altri potrebbe venir spostato o, addirittura, inglobato in una struttura più grande).

Il secondo caso riguarda "*l'essere contenuto in una particolare area*": tale aspetto influenza il processo di derivazione in quanto permette di caratterizzare in maniera migliore l'importanza degli oggetti. Di conseguenza, è possibile determinare quali oggetti debbano essere rappresentati sulla mappa, così come la loro modalità di rappresentazione (nella generalizzazione degli edifici, ad esempio, si recuperano informazioni sulla tipologia della partizione di appartenenza, che può essere "urbana" piuttosto che "rurale").

3.2.3 Soluzioni proposte

Per il processo di partizionamento si è scelto di non prendere in considerazione la strategia dei "tile intelligenti", in cui differenti categorie di geometrie ven-

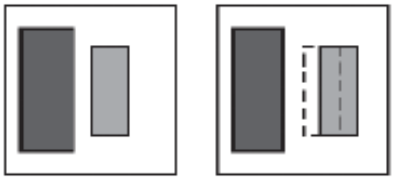

Figure 3.10: Displacement: indica il movimento di un oggetto, mantenendo la forma invariata.

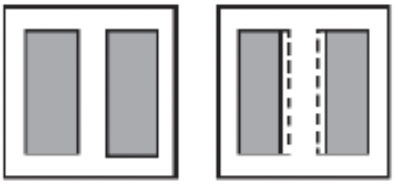

Figure 3.11: Enlargement. indica un globale incremento della forma di un oggetto.

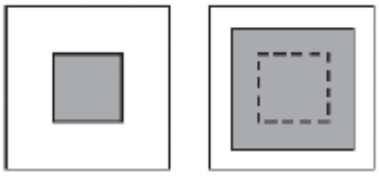

Figure 3.12: Exaggeration: definisce un locale incremento (o diminuzione) di un oggetto, la sua forma si modifica. Si usa quando, a causa della riduzione di scala, elementi importanti non risultano più leggibili e chiari.

gono combinate tra loro per formare partizioni chiuse [5], puntando invece alla creazione di una griglia di tile rettangolari: tale scelta si traduce in una maggiore semplicità nella gestione delle partizioni, andando a risolvere i conflitti e le criticità in un secondo momento, aumentando il dataset con geometrie provenienti da partizioni vicine.

La fase iniziale di analisi si è quindi spostata sull'individuazione di possibili metodologie per l'integrazione di geometrie di partizioni "vicine" in quella esaminata.

Per quanto riguarda i casi continui, si è visto che nei grafi un semplice inseguimento senza alcun controllo non avrebbe portato ad una soluzione ottimale, in quanto nella maggior parte dei casi sarebbe caricata in memoria un'eccessiva quantità di dati, a causa della stretta connessione che sussiste tra le geometrie della rete fluviale o anche stradale (che si dirama anche per lunghi tratti). In figura 2.13 si osserva che, inseguendo le geometrie senza alcun tipo di controllo a partire dalla partizione in alto a sinistra, si andrebbe a caricare in memoria la quasi totalità delle geometrie del dataset globale (grafo in rosso).

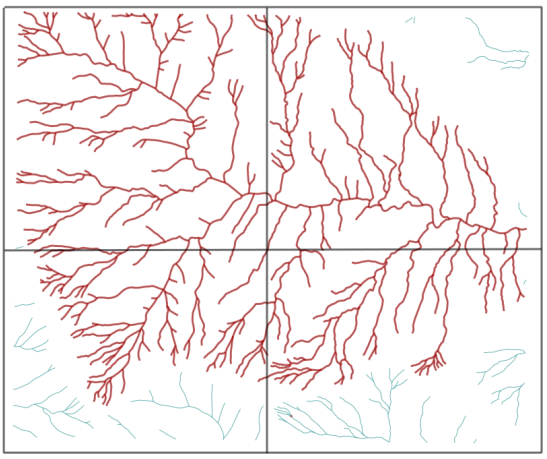

Figure 3.13: Inseguimento senza controllo su grafo: troppe geometrie vengono integrate nella partizione

Ecco, quindi, l'importanza di definire delle opportune condizioni di termine (ad esempio la lunghezza del tratto da inseguire), che portino ad integrare il dataset con un numero adeguato di geometrie (in rosso il nuovo grafo, con un numero di geometrie selezionate più consono).

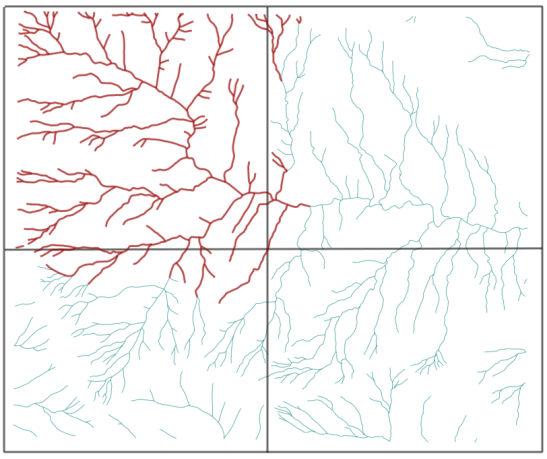

Figure 3.14: Inseguimento effettuato fissando delle opportune condizioni di termine.

Per i *casi discreti*, invece, il termine "inseguimento" sarebbe improprio in quanto una geometrie termina con se stessa, poiché essa non è parte di alcuna struttura (come, ad esempio, avviene per i grafi). In questo caso l'ampliamento del dataset di lavoro avviene attraverso la creazione di buffer su quegli elementi prossimi al bordo, in modo da integrare quelle geometrie esterne che presentano una qualche relazione con quelle interne.

L'importanza di fissare delle opportune condizioni di termine si ritrova anche nell'analisi dei casi discreti: continuare ad iterare la creazione di buffer finché non si termina di aggiungere nuovi elementi funziona solamente per alcuni tipi di geometrie, in particolare per quelle di piccole dimensioni (come, ad esempio, per gli edifici – figura 2.15).

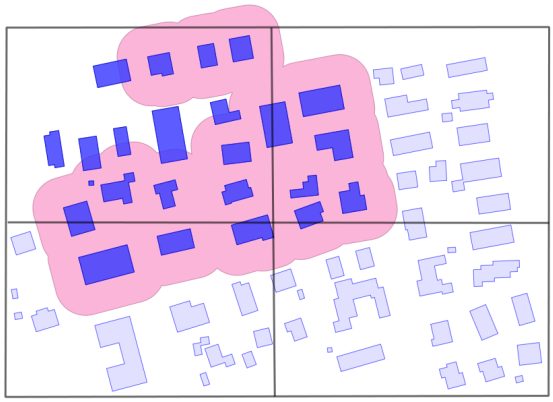

Figure 3.15: Integrazione nel tile di nuove geometrie tramite buffer.

Per geometrie di medie/grandi dimensioni (come, ad esempio, i boschi), si potrebbero avere casi in cui gli elementi aggiunti hanno dimensioni ragguardevoli, facendo perdere il controllo sulla partizione su cui si sta lavorando, pregiudicando quindi il processo di partizionamento che prevede di caricare in memoria dataset di dimensioni ragionevoli. In figura 2.16 si vede chiaramente come, applicando il concetto appena visto di iterazione del buffer degli elementi prossimi al bordo, la partizione in alto a sinistra si troverebbe a dover gestire geometrie la cui dimensione è più del doppio di quella del tile stesso.

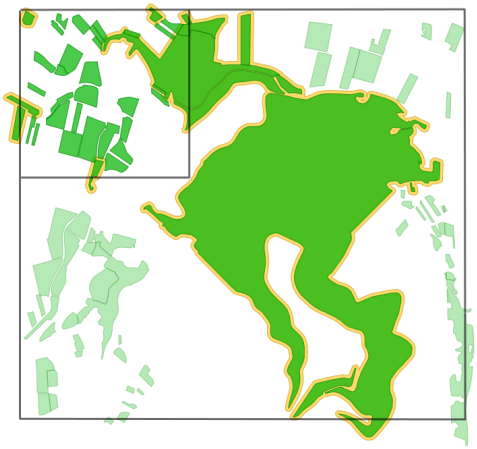

Figure 3.16: Problematica dell'integrazione di grandi geometrie.

Ecco che, in caso di grandi geometrie, risulta più efficace spezzare l'elemento sul bordo del tile, prevedendo una successiva gestione che garantisca l'integrità e unicità del dato finale.

Nel prossimo capitolo si andranno ad esaminare le metodologie che gestiscono il processo di partizionamento del dataset globale per i vari algoritmi di generalizzazione presenti nel progetto CARGEN.

Chapter 4

Tile Manager for Cartographic Generalization

4.1 Il partitioning nel progetto CARGEN

Attualmente, all'interno del progetto CARGEN, si fa un uso intenso del DB in cui il dataset iniziale è memorizzato: lavorando con tabelle "lato client" si ha la possibilità di rendere i vari algoritmi indipendenti dal DB utilizzato e più efficienti, sfruttando la potenza computazionale del calcolatore (implementando politiche di gestione del rischio di sovraccarico della memoria).

4.1.1 Considerazioni iniziali

Tra le varie soluzioni proposte negli anni, sono quelle relative alla necessità di definire in maniera ottimale il "contesto spaziale" ad aver animato principalmente il lavoro di questa tesi: l'impossibilità, quindi, di considerare un oggetto da generalizzare come entità unica e scollegata dall'ambiente in cui si trova [16]. Questo significa considerare l'ambiente in funzione di aspetti quali prossimità, distanza o partecipazione in strutture più grandi.

L'idea di fondo è che ciascuna partizione ha al suo interno un sottoinsieme di dati sui quali si riesce a portare a termine correttamente la generalizzazione: questa zona coincide con la parte centrale del tile, che ha a disposizione un'informazione spaziale certa (*validated area*).

Figure 4.1: L'informazione spaziale a disposizione diminuisce man mano che ci si avvicina al bordo della partizione

Come si vede dalla figura, man mano che ci si allontana dal centro della partizione l'informazione spaziale diminuisce progressivamente, in quanto un tile non ha la visibilità di tutto ciò che si trova esternamente ad esso, rendendo impossibile la validazione di quelle geometrie che si trovano in prossimità del bordo. Integrando il dataset della partizione con alcune geometrie vicine, aumentano le dimensioni della validated area sino a ricoprire interamente la partizione di partenza, che pertanto verrà generalizzata correttamente.

Le problematiche che sorgono con questo approccio riguardano principalmente:

1. la modalità di integrazione di nuove geometrie in una partizione, ossia individuare – ove necessario - delle "condizioni di termine" che permettano all'algoritmo di comprendere quale debba essere l'informazione minima, necessaria e sufficiente per garantire una corretta generalizzazione di ciascuna partizione, e in particolar modo di quelle geometrie presenti in prossimità dei bordi, mantenendo la continuità dell'informazione spaziale di partizioni adiacenti nella mappa risultante.

2. la successiva gestione delle ambiguità, ossia di quelle geometrie che si trovano a cavallo di più partizioni, e che verrebbero generalizzate più di una volta (presenza di duplicati nel risultato finale).

4.1.2 Analisi situazione di partenza

Gli attuali algoritmi di generalizzazione presenti nel progetto CARGEN operano su:

- Rete fluviale
- Edifici
- Rete stradale
- Rete ferroviaria
- Scoline
- Grandi aree
- Geometrie puntuali

A partire da tali algoritmi si è quindi cercato di trovare una loro possibile classificazione in funzione delle problematiche che espongono al tema del partitioning. Dopo aver appurato che le geometrie puntuali non presentano particolari criticità, prendendo spunto dalla suddivisione tra casi continui e discreti

osservata nel capitolo precedente (*2.3.1 Soluzioni proposte*), sono state individuate tre grandi classi in grado di fornire una metodologia generale e adattabile ai vari casi esistenti: *grafi, piccole geometrie e grandi geometrie*.

1) Grafi

In questa classe rientrano tutti i casi continui, ossia le generalizzazioni di fiumi, strade e rete ferroviaria. L'aggiunta di nuove geometrie nel dataset della partizione deve essere mediata da specifiche condizioni di termine, per evitare l'aggiunta di un numero eccessivo di geometrie: il processo di inseguimento, quindi, prevede di fermarsi una volta raggiunto il Minimo Set di Informazione per la Generalizzazione (MSIG).

Per quanto riguarda la rete stradale si è visto che una buona estensione può essere rappresentata da un inseguimento di 3km, che è la lunghezza minima per identificare un tratto stradale come "tratto autostradale". Con tale estensione, inoltre, si riuscirebbe a risolvere la problematica degli incroci, i quali hanno la necessità di essere trattati nella loro interezza, come un'unica entità, concetto che viene minato dalla creazione delle partizioni.

Il caso dei fiumi lo si vedrà in particolare nel prossimo capitolo.

Le ambiguità che si generano, pertanto, non riguardano tanto la presenza di geometrie duplicate nel risultato finale, bensì casi di situazioni incomplete: in figura si nota come un tratto di fiume situato tra due partizioni sia in realtà formato da due geometrie semi-sovrapposte, che derivano dalla generalizzazione di entrambe le partizioni coinvolte.

Figure 4.2: Grafi: geometrie sul bordo, generalizzate da due partizioni adiacenti, si sovrappongono.

2) Piccole geometrie

Questa classe racchiude i casi discreti di dimensioni ridotte, che riguardano quindi la generalizzazione degli edifici e delle scoline. L'integrazione di nuove geometrie nella partizione avviene attraverso un buffer attorno a quelle presenti in prossimità del bordo, iterandolo fintanto che esistono geometrie toccate.
Questo approccio, valido sia per le scoline che per gli edifici, permette di mantenere omogenea la generalizzazione di gruppi di geometrie che presentano una relazione di dipendenza: in questo modo un cluster di scoline (così come un gruppo di edifici) che si trova in prossimità del bordo viene generalizzato come un'unica entità. Con questo approccio, le situazioni ambigue saranno effettivamente geometrie complete e, supponendo che l'algoritmo sia robusto e deterministico, si dovrà - come si evince dalla figura - prevedere la gestione di duplicati.

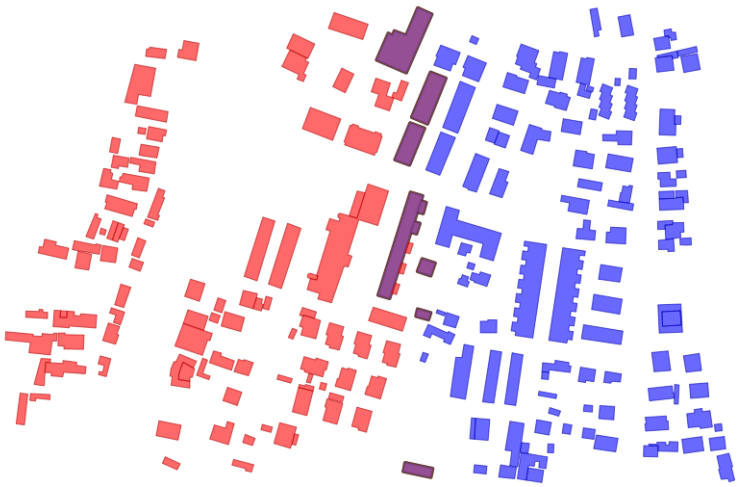

Figure 4.3: Piccole geometrie: se presenti sul bordo, generalizzate in precedenza da due partizioni adiacenti, vengono a coincidere

3) Grandi geometrie

L'ultima classe che chiude questa classificazione degli algoritmi di derivazione racchiude al suo interno quei casi discreti di grandi dimensioni (come ad esempio boschi, laghi, ecc...).

Il problema derivante dalla gestione di questi elementi è rappresentato dal fatto che vi è il rischio di ritrovarsi a gestire geometrie di dimensioni maggiori a quelle del tile stesso. La fase di integrazione, quindi, viene sostituita con quella di "divisione" di tali elementi, che porterà pertanto a generare delle situazioni incomplete in prossimità dei bordi. La loro gestione si basa quindi sulla fusione

delle geometrie che si ritrovano ad essere complementari, in quanto parti della stessa grande geometria iniziale.

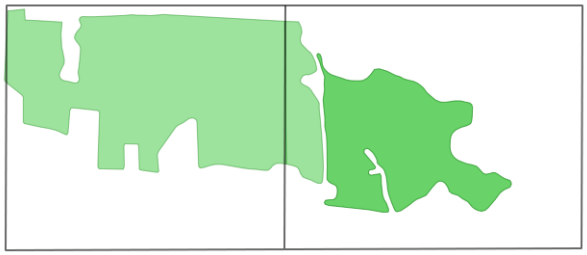

Figure 4.4: Grandi geometrie: se presenti sul bordo, generalizzate in precedenza da due partizioni adiacenti, appartengono alla stessa geometria iniziale

4.2 Tile Manager

La strada intrapresa è stata quella di implementare un Tile Manager (TM) per gestire l'intero processo di partizionamento, guidando la generalizzazione di ogni partizione e fornendo alcuni metodi che risolvessero situazioni particolari.

4.2.1 Gestione delle partizioni

TM e algoritmo di generalizzazione dialogano tra loro lungo tutto il processo:

1. il TM genera n partizioni rettangolari del dataset globale, e su ciascuna lancia l'algoritmo;
2. ciascun algoritmo esegue una fase di pre-processing in cui identifica quelle particolari situazioni che potrebbero generare dati incompleti in output, richiedendo al TM geometrie aggiuntive secondo la modalità di estensione più adatta (inseguimento, buffer...)
3. ciascun algoritmo, avendo ora a disposizione l'informazione necessaria per completare la generalizzazione, inizia a processare il dataset in input (indipendentemente dalle altre partizioni)
4. una volta terminata l'esecuzione di ogni algoritmo, il TM controlla i risultati ottenuti eseguendo un'intersezione con il rispettivo tile originale:
    - tutte le geometrie che fanno parte di situazioni interne ad esso vengono salvate;
    - le geometrie che fanno parte di situazioni sul bordo sono considerate ambigue, e vengono pertanto inserite in un bucket per l'integrazione finale. Da sottolineare il fatto che in tale bucket sono presenti delle situazioni, e non necessariamente geometrie generalizzate (si pensi

al caso di un cluster di scoline, o ad un raggruppamento di edifici): essenziale è quindi mantenere traccia di tutte quelle geometrie collegate a ciascuna situazione.

Alcune situazioni, inoltre, potrebbero risiedere sul bordo del dataset globale: in questi casi non si può parlare di ambiguità, pertanto le si andrà a salvare nel risultato finale.

Dopo aver processato l'intero dataset il TM si ritrova con n partizioni completamente generalizzate, eccezion fatta per tutte quelle situazioni presenti sui bordi che sono finite nel bucket.

Terminata l'esecuzione sulle partizioni, quindi, il TM inizia l'analisi delle ambiguità presenti nel bucket: nel caso di grafi coincidono con le effettive geometrie derivate, ma in generale possono rappresentare anche più di una geometria. A seconda del tipo di generalizzazione applicata, il TM lascerà risolvere le ambiguità all'algoritmo che le ha generate (e che pertanto conosce la loro struttura), a seconda che si tratti di grafi, piccoli o grandi geometrie: qualora fosse necessario, quest'ultimo avrebbe ora la possibilità di accedere al resto dei dati già processati e salvati.

4.2.2 Implementazione

In questo lavoro si è cercato, per quanto possibile, di non stravolgere gli algoritmi già esistenti, potendoli quindi riutilizzate nella loro quasi totalità: minime aggiunte al codice esistente, più alcuni nuovi metodi aggiuntivi, senza intaccare il processo originale che lavora su un unico dataset.

Le diversi classi di algoritmi che intendono sviluppare la gestione del partizionamento dovranno necessariamente implementare l'interfaccia Parallelization, che fornisce alcuni metodi aggiuntivi per la gestione del tiling.

public void setConnection(Connection vconn)
Ciascuna partizione lavorerà su tabelle presenti in memoria, quindi l'algoritmo dovrà impostare la propria connessione come virtuale, e non verso il DB.

textttpublic String[] getInputTables(), public String[] getOutputTables()
L'algoritmo deve dichiarare di quali tabelle necessita, in quanto il TM deve sapere quali tabelle dovrà tagliare. Allo stesso modo dovrà conoscere il nome delle tabelle in output da salvare e/o successivamente gestire, compresa anche l'eventuale tabella delle situazioni (che può coincidere, comunque, con l'effettiva tabella finale).

```
public void manageBoundary(TileManager tm, String[] tabin, Geometry
boundary)
```

Questo metodo permette all'algoritmo di controllare tutte quelle geometrie (o situazioni) presenti in prossimità del bordo che potrebbero introdurre delle criticità, potendo decidere di volta in volta come comportarsi: dopo averle individuate, quindi, chiamerà il TM per avere dati aggiuntivi che risolvano potenziali situazioni problematiche, attraverso la modalità più adatta di estensione tra quelle elencate in precedenza (inseguimento, buffer statico...).

Al suo interno è prevista un'ulteriore chiamata al TM, per controllare che eventuali situazioni problematiche non risiedano sull'effettivo "bordo del mondo" (ossia dell'originale dataset globale), fatto che non le etichetterebbe più come "problematiche".

`public void setBoundary(Geometry boundary)`
`public Geometry getBoundary()`
Metodi che, rispettivamente, inizializzano e recuperano l'informazione spaziale relativa al bordo della partizione.

`public void runParallel()`
E' il metodo che lancia l'esecuzione dell'algoritmo di generalizzazione su una particolare partizione.

`public void solvePendings(Table bucket, String tablePending, Connection dbConn)`
Questo metodo gestisce l'analisi delle situazioni ambigue presenti nella tabella bucket, andando di volta in volta a risolvere tali ambiguità scegliendo la modalità più opportuna – a seconda che si tratti di grafi, piccole o medie geometrie – per scegliere quali geometrie pending scrivere nella tabella finale.

`public void writeInResult(int id, String tableName, Connection conn)`
La relazione che sussiste tra una situazione e le geometrie generalizzate ad essa associate è presente nella definizione della tabella `tabSituations`: questo metodo, perciò, effettua la scrittura sul DB delle geometrie risultanti.

`public void writeInPending(int id, String tableName, Connection conn)`
Per questo metodo valgono le stesse considerazioni fatte per il precedente: in questo caso, però, ad essere scritte non sono le geometrie finali bensì quelle in attesa di validazione, che saranno risolte nella fase finale di analisi delle ambiguità (geometrie pending). Un'ulteriore differenza dal metodo precedente deriva dal fatto che la scrittura sul DB viene effettuata in maniera esclusiva, attraverso alcune primitive di lock-and-wait, a causa del parallelismo che sussiste tra i processi che lavorano sulle partizioni (*tema affrontato nel capitolo 5*).

L'unica struttura dati aggiuntiva che l'algoritmo di generalizzazione dovrà implementare sarà quella della "tabella delle situazioni" (`tabSituations`), che

conterrà tre attributi:

1. Id – attributo numerico che identifica la particolare situazione;
2. Geometry – attributo spaziale;
3. Related – attributo alfanumerico, composto da una sequenza di id separati da un ";" che identificano le geometrie generalizzate che compongono tale situazione (nel caso di grafi i campi Id e Related avranno il medesimo valore).

Il TM, dal canto suo, espone alcuni metodi per l'integrazione di nuove geometrie nel dataset iniziale:

`public Collection<Feature> followGeom(Geometry geom, Geometry boundary, String table, double maxLength, int buffer, String mask)`
Questo metodo restituisce una collezione di Feature che rappresenta il sottografo che si estende all'esterno della partizione corrente a partire dalla geometrie "geom", per una lunghezza definita dal parametro maxLength. E' possibile arricchire tale estensione con parametri quali l'eventuale buffer da utilizzare e la relazione topologica da utilizzare (rispettivamente con i parametri buffer e mask).

`public Collection<Feature> staticBuffer(Geometry geom, Geometry boundary, String table, int buffer, String mask)`
Questo metodo restituisce una collezione di Feature formata da tutte quelle geometrie esterne alla partizione che sono toccate dal buffer sulla geometrie geom.

`public Collection<Feature> iterateBuffer(Geometry geom, Geometry boundary, String table, int buffer, String mask)`
A differenza del metodo precedente, l'aggiunta di Feature viene iterata ciclicamente eseguendo un ulteriore buffer sulle geometrie appena aggiunte, terminando solamente quando non ci sono effettivamente più elementi da integrare.

`public boolean isOnGlobalBoundary(Geometry geom)`
Metodo che controlla se una particolare geometria risiede o meno sul bordo del dataset globale.

Chapter 5

Alcuni esempi concreti

Dopo aver definito a livello macroscopico il processo di partizionamento, illustrando la sequenza di passi eseguiti dal TileManger e dall'algoritmo di derivazione, ci si è concentrati nella risoluzione effettiva di uno dei casi elencati nella classificazione iniziale: la Generalizzazione della Rete Fluviale.

5.1 Partizionamento della rete fluviale

La generalizzazione della rete fluviale prevede un algoritmo di selezione: ciò significa che, una volta partizionato il dataset, non sarà necessario preservare la dipendenza gerarchica tra due tratti fluviali (ossia tra un corso principale e i suoi affluenti): ciò che rende corretto il processo di derivazione di un tratto fluviale è solamente la sua effettiva presenza nella mappa risultante.

5.1.1 La generalizzazione della rete fluviale

L'algoritmo presente nel progetto CARGEN si sviluppa nei seguenti passi:

- Ricostruzione della direzione del flusso;
- Data enrichment;
- Ricostruzione dei corsi fluviali;
- Eliminazione dei tratti fluviali troppo corti;
- Eliminazione dei tratti fluviali per sfoltire aree tropo dense.

Il primo passo non presenta particolari criticità al partizionamento, in quanto si occupa solamente della ricostruzione di un fiume a partire dai suoi tratti per conoscerne la direzione. Questa fase è fondamentale in quanto permette di asserire se due sezioni convergono in una o se, al contrario, una singola si dirama in due.

Il secondo passo, attraverso una procedura di tipo top-down (partendo dalla sorgente sino alla foce), assegna a ciascun tratto delle informazioni aggiuntive utili alla ricostruzione, ossia:

- S(i) : l'ordine di Strahler dell'i-esimo tratto;
- L(i) : la distanza alla sorgente più lontana;
- B(i) : il numero totale di diramazioni.

A differenza della precedente, questa fase introduce alcune criticità, in quanto tali informazioni sono strettamente legate al contesto spaziale, in quanto fanno riferimento al fiume nella sua interezza. Un suo partizionamento, perciò, andrebbe a falsare questi parametri, portando successivamente ad una ricostruzione dei fiumi potenzialmente differente, e a considerazioni errate in riferimento alla lunghezza dei fiumi.

La problematiche in questione, tuttavia, vengono risolte dalla preventiva integrazione del dataset di tutte quelle geometrie ottenute attraverso l'inseguimento del grafo all'esterno del tile. In questo modo i parametri, nonostante risultino differenti da quelli che si avrebbero con un'unica generalizzazione globale, raggiungono la soglia minima per garantire la validazione del processo di selezione.

La ricostruzione dei corsi fluviali, quindi, procede senza particolari problematiche: a partire dalla "foce" con l'ordine di Strahler maggiore l'algoritmo ricostruisce il fiume sino alla sorgente scegliendo come prossimo lato da esaminare, di volta in volta, uno dei padri del tratto attuale secondo considerazioni inerenti ai parametri S-L-B.

Poiché tratti fluviali di lunghezza inferiore alla soglia di 250m vengono scartati, un'eliminazione di tali geometrie presenti sul bordo rischierebbe di introdurre potenziali errori, poiché un fiume potrebbe continuare all'esterno del tile per più di tale soglia. Inseguire le geometrie sul bordo all'esterno per 250m – come già detto in precedenza – risolve la problematica di questa fase.

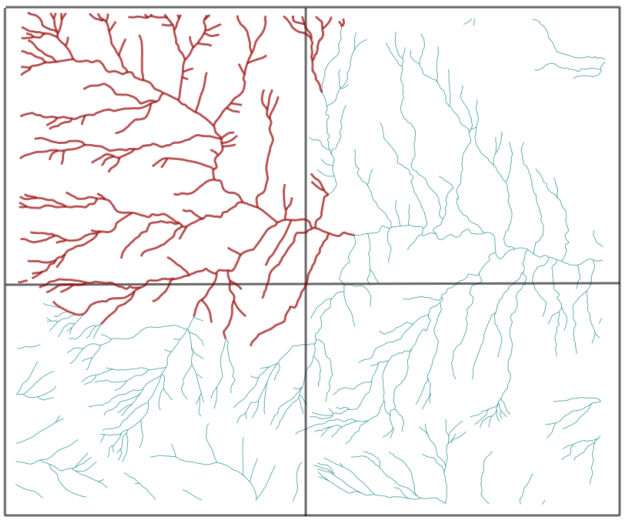

Figure 5.1: Estensione della partizione tramite inseguimento "intelligente"

L'ultima fase, tuttavia, nonostante gli accorgimenti appena citati, presenta ancora una criticità importante: per il calcolo della densità, infatti, si utilizzano dei buffer attorno ai tratti di fiumi che permettono di calcolare la percentuale di area in cui essi si sovrappongono. La mancata conoscenza della situazione "esterna" alla partizione comporta la creazione di un risultato alterato, poiché non tiene conto degli apporti che dovrebbero arrivare dai tratti di fiumi esterni alla partizione. Nell'esempio sottostante, infatti, il buffer del tratto di fiume evidenziato in rosso dovrebbe nella realtà intersecare i buffer azzurri, ma con l'introduzione del partizionamento ciò non sarebbe più valido.

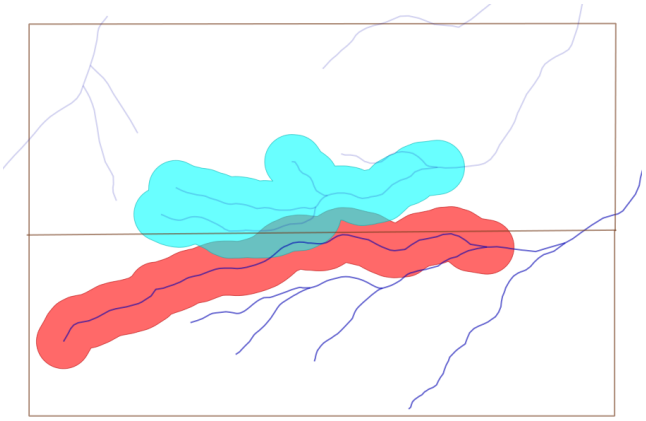

Figure 5.2: Problematica della generalizzazione fluviale: i buffer di fiumi esterni alla partizione non vengono presi in considerazione

In questo caso, perciò, nella fase di integrazione è necessario tener conto anche di questi buffer. L'inseguimento, pertanto, non sarà semplice, bensì effet-

tuato con l'ulteriore ausilio di un buffer , e in egual modo verranno caricate nel dataset anche quelle geometrie esterne il cui buffer interseca il bordo del tile.

5.1.2 Gestione del processo di partizionamento

L'approccio di questo lavoro è stato quello di implementare un plugin in OpenJump per potersi ricollegare ai plugin già sviluppati, al fine di avere un immediato riscontro visivo di quanto prodotto dai processi di partizionamento e generalizzazione.

Una semplice schermata iniziale richiede all'utente di scegliere la tipologia di algoritmo da eseguire, e il numero di partizioni che intende creare (attraverso la definizione del numero di righe e colonne). La schermata successiva presenta, quindi, il plugin relativo alla generalizzazione scelta (riusabilità), che lancerà l'esecuzione del TileManager.

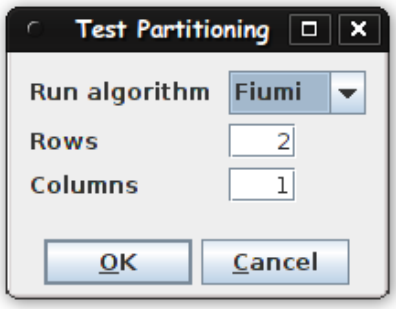

Figure 5.3: OpenJump: Maschera per la scelta del tipo di algoritmo di generalizzazione, e del numero di partizioni

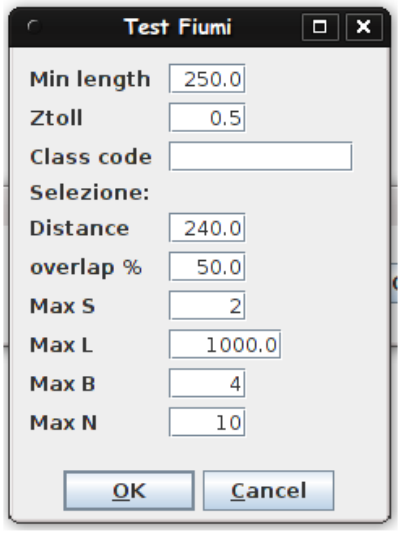

Figure 5.4: OpenJump: Maschera dell'algoritmo originale di generalizzazione fluviale.

Riassumendo i passi dell'intero processo:

Listing 5.1: I passi del processo associato al TileManager

```
1) il TM genera n partizioni rettangolari del dataset
   globale e su ciascuna lancia l'algoritmo;

2) ciascuna partizione esegue una fase di pre-processing
   per identificare  situazioni potenzialmente
   problematiche, e successiva integrazione del dataset;

3) inizio della generalizzazione di ogni partizione;

4) al termine dell'esecuzione di ciascuna partizione, il
   TM  controlla i risultati ottenuti eseguendo
   un'intersezione con il rispettivo tile originale:
        a) tutte le geometrie che fanno parte di
           situazioni interne ad esso vengono salvate;
        b) le geometrie che fanno parte di situazioni sul
           bordo sono considerate ambigue, e vengono
           pertanto inserite in un bucket per
           l'integrazione finale.

5) una volta che tutte le partizioni hanno portato a
   termine la loro esecuzione, il TM lascia all'algoritmo
   di generalizzazione l'analisi e la risoluzione delle
   situazioni ambigue.
```

1) il TM genera n partizioni rettangolari del dataset globale, e su ciascuna lancia l'algoritmo

Attraverso una semplice query spaziale il TM crea n partizioni a partire dal dataset globale.

In questa fase, inoltre, si crea l'informazione relativa al perimetro della partizione, che servirà nelle fasi successive per tutte quelle considerazioni inerenti alle geometrie che intersecano il bordo.

2) ciascun partizione esegue una fase di pre-processing per identificare situazioni potenzialmente problematiche, e successiva integrazione del dataset

Ogni algoritmo di generalizzazione, attraverso il metodo `manageBoundary(TileManager tm, String[] tabin, Geometry boundary)`, analizza e gestisce le situazioni potenzialmente critiche secondo la modalità più congeniale al tipo di processo.

Per quanto riguarda i fiumi, quindi, si opererà un controllo su tutte le geometrie del dataset che intersecano il bordo della partizione, e si andrà ad operare

un "inseguimento con buffer".

I parametri da utilizzare per tale inseguimento arrivano direttamente dagli studi effettuati per l'implementazione dell'algoritmo di generalizzazione: P = 120m (valore del buffer attorno ai fiumi per il calcolo della densità) L = 1000m (lunghezza minima per la fase "eliminazione tratti fluviali in aree dense")

La soglia dei 250m relativa alla fase *eliminazione tratti fluviali troppo corti* viene scartata in quanto non consentirebbe di gestire l'*eliminazione per densità*.

3) inizio della generalizzazione di ogni partizione

Questa fase coincide con l'effettiva esecuzione dell'algoritmo di generalizzazione dei fiumi.

A differenza del normale processo, ora si tiene conto del nuovo concetto introdotto dalle "situazioni", creandone la relativa tabella (`tabSituations` – vedi paragrafo *3.2.1 Implementazione*). Ad ogni modo, per questa classe le situazioni coincidono effettivamente con le geometrie generalizzate, quindi ogni ID identificativo della situazione coinciderà con l'unico id presente nel campo RELATED.

E' necessario, inoltre, apportare un'ulteriore modifica: nel processo originale le geometrie sul bordo vengono automaticamente salvate senza passare per la fasi di pruning, mentre ora si rende necassario estendere i controlli di lunghezza e densità anche a suddetti elementi.

4) al termine dell'esecuzione di ciascuna partizione, il TM controlla i risultati ottenuti eseguendo un'intersezione con il rispettivo tile originale

Terminata la generalizzazione della partizione, il TM effettua una query spaziale sulla tabella delle situazioni creata dall'algoritmo, salvando tutte quelle interne al tile ed inserendo in un bucket quelle che intersercano il bordo, affinché vengano successivamente analizzate nella fase di integrazione finale.

Le geometrie collegate a situazioni ambigue, quindi, vengono anch'esse salvate sul DB in un'altra tabella (`tabPending`), in attesa della loro scrittura finale. Rispetto al dataset globale questa tabella presenta un numero di geometrie in proporzione nettamente inferiore, pertanto si è deciso di utilizzarne una unica per tutte le partizioni, sfruttando la modalità di scrittura *lock-and-wait* del metodo esposto dal TM public void writeInPending(int id, String tableName, Connection conn). In questo modo, in fase di creazione del bucket (che presenta lo stesso schema di tabSituations) viene garantita la corrispondenza tra "id della situazione" e "id delle geometrie pending".

5) una volta che tutte le partizioni hanno portato a termine la loro esecuzione, il TM lascia all'algoritmo di generalizzazione l'analisi e la risoluzione delle situazioni ambigue

Una volta collezionate tutte le situazioni ambigue nella tabella bucket, il TM crea una nuova istanza dell'algoritmo di generalizzazione per lanciare il metodo `solvePendings(Table bucket, String tablePending, Connection dbConn)`, che provvede all'analisi e risoluzione delle ambiguità. Per quanto riguarda i fiumi, essendo le situazioni in realtà delle effettive geometrie generalizzate, tale funzione si riduce alla fusione di quegli elementi presenti nella tabPendings che da un punto di vista spaziale si sovrappongono, andando a scrivere il nuovo elemento creato nella tabella finale.

5.1.3 Analisi dei risultati

Gli obiettivi prefissati all'inizio del lavoro sono stati, in prim'ordine, la corretta gestione della memoria al fine di portare a termine correttamente il processo di generalizzazione di un qualsiasi tipo di dataset, con un'attenzione particolare per quelli di grandi dimensioni.

In secondo luogo, si è cercato di proporre una soluzione che non introducesse un eccessivo degrado nelle performance del processo di generalizzazione, puntando a preservare – se non addirittura migliorare – le tempistiche d'esecuzione dell'intero processo.

Per avere un parametro di riferimento si è eseguita la versione originale dell'algoritmo, prendendo come dati in input geometrie provenienti da una tabella in un database ORACLE 10g di circa 11,000 elementi.

Successivamente sono state effettuate alcune prove eseguendo l'algoritmo di partizionamento, in particolare creando 4 tile e generalizzandone solo due di essi contemporaneamente (2 processi attivi in parallelo). Dopo varie prove si è notato come le performance, con queste impostazioni, degradassero di quasi il 50%: per questo motivo si è provata una nuova configurazione con un numero di processi attivi pari all'effettivo numero di partizioni. Notando un lieve miglioramento, le prove si sono via via ripetute incrementando di volta in volta il numero di partizioni (e di conseguenza il numero di processi attivi in parallelo), evidenziando un'alta scalabilità dell'algoritmo, che con più di 25 partizioni ottiene tempistiche migliori della versione originale (tabella).

Questo risultato si spiega alla luce dell'iniziale partizionamento del dataset: questa fase risulta essere la più critica a livello computazionale, in quanto una query spaziale per ricavare il tile su cui lavorare richiede comunque del tempo. Un'alta parallelizzazione di questa fase permette di guadagnare quel tempo che una partizione dovrebbe attendere qualora venisse processata in maniera sequenziale (ossia nel caso in cui il numero di processi sia minore del numero di partizioni), beneficiando quindi del minor tempo d'esecuzione della generalizzazione di dataset di dimensioni inferiori.

CLASSE	MODALITA'	TEMPI D'ESECUZIONE
FIUMI Originale	1 Partizione	1' 45"
	1 Processo	
FIUMI Parallelo	4 Partizioni	2' 58"
	2 Processi	
	4 Partizioni	2' 53"
	4 Processi	
	9 Partizioni	2' 36"
	9 Processi	
	16 Partizioni	2' 05"
	16 Processi	
	25 Partizioni	1' 51"
	25 Processi	
	36 Partizioni	1' 42"
	36 Processi	
	40 Partizioni	1' 27"
	40 Processi	

Table 5.1: Generalizzazione fluviale: Tabella dei risultati

Ad ogni modo, questo può essere ritenuto un buon risultato qualora si stesse processando un dataset di dimensioni contenute, in quanto avere un numero di processi uguale al numero di partizioni significa di fatto caricare in memoria l'intera mappa: nel caso di dataset di grandi dimensioni, quindi, ciò porterebbe ad un inevitabile stallo della macchina qualora tali dimensioni superassero quelle della memoria gestita dal calcolatore.

5.2 Partizionamento delle maglie stradali

Una menzione a parte merita la trattazione delle maglie stradali. Nonostante non rientri tra i casi di "generalizzazione", è comunque uno tra i passi fondamentali di alcuni algoritmi di derivazione, come ad esempio "Edifici" e "Strade". La particolarità di questo algoritmo è che non rientra in nessuno dei casi visti in precedenza, in quanto i dati in input appartengono ad una classe differente rispetto ai dati in output: partendo dalle strade si ottengono delle maglie.

Il concetto su cui si basa la creazione di maglie su un dataset preventivamente partizionato, ad ogni modo, risulta simile a quello già elencato per i vari algoritmi di generalizzazione: la parte centrale di una partizione risulta essere una zona certa di cui si riescono a validare le geometrie in output, mentre

l'incertezza aumenta avvicinandosi al bordo (figura).

Figure 5.5: L'informazione spaziale a disposizione diminuisce man mano che ci si avvicina al bordo della partizione

Seguendo questo principio, tutte le maglie interne alla partizioni vengono quindi individuate correttamente e con precisione, mentre quelle che risiedono sul bordo del tile risultano non-ricostruibili in quanto perdono l'informazione che arriva dalle strade che le compongono ma che si ritrovano all'esterno della partizione.

Per le maglie non c'è modo di conoscere preventivamente la lunghezza dei tratti stradali che le compongono: per risolvere questa criticità si è osservato come, in realtà, le maglie che intersecano i bordi delle partizioni possono essere identificate sfruttando proprio l'insieme dei tratti stradali che rimangono una volta che si sono tolti quelli appartenenti a maglie già trovate (in figura vengono indicate in rosso le maglie trovate durante la successiva fase d'analisi, in cui si eliminano le strade di maglie già trovate).

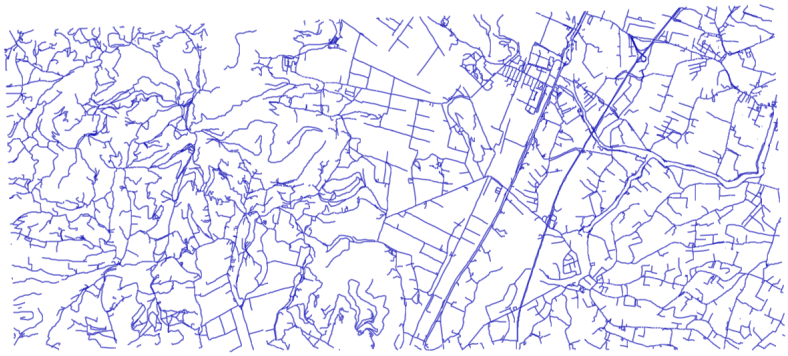

Figure 5.6: Maglie stradali: Prima

Per riuscire ad implementare questa soluzione, si è leggermente modificato il concetto di "situazioni".

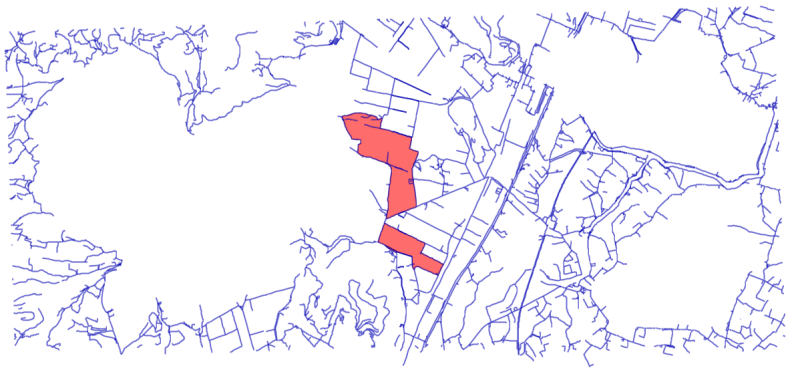

Figure 5.7: Maglie stradali: Dopo

La `tabSituations`, in questo caso, contiene sia le maglie calcolate all'interno della partizione, sia le strade che rimangono dopo aver eliminato tutti i tratti stradali che compongono le maglie.

Il metodo `writeInResult(int id, String tableName, Connection conn)`, in questo caso, controlla che la geometria da salvare sia un Poligono e non una LineString, poiché in quest'ultimo caso si tratta di una strada utile nella successiva fase di risoluzione dei casi pending, e quindi da salvare nella `tabPendings`. Il metodo writeInPending(int id, String tableName, Connection conn), invece, scrive com'è giusto che sia i tratti stradali nella `tabPendings`.

In questo modo, il `solvePendings` si ritrova a dover eseguire nuovamente l'algoritmo di creazione delle maglie sul dataset formato dale rimanenti strade, salvando quindi solamente quelle presenti sul bordi della griglia create dal TM all'inizio del partizionamento.

Chapter 6

Parallelizzazione

Un'ultima considerazione è stata fatta sul possibile approccio "parallelo" al processo di partizionamento. Si trattava di capire come gestire le geometrie "esterne" a ciascuna partizione:

1. aggiungerle al dataset della partizione, così da non essere più ricalcolate in futuro;
2. usarle per la generalizzazione del dataset della partizione, senza effettivamente salvarle in maniera permanente.

La strada intrapresa, come illustrato nei capitoli precedenti, è stata quella di rendere indipendenti le partizioni, integrando nel dataset di ciascuna partizione nuove geometrie esterne senza, però, salvarle in maniera indistinta nella tabella finale.

I vantaggi di un partitioning serializzato coinvolgono solamente l'aspetto della miglior gestione della memoria: un partitioning parallelo, invece, in alcuni casi consente di non degradare eccessivamente le performance sotto l'aspetto temporale, potendo generalizzare più partizione contemporaneamente.

6.1 Le principali architetture parallele

Le tre più comuni architetture parallele sono: SMP, Cluster ed Ibrida.

a) SMP – Shared Memory Parallel

Nell'architettura SMP due o più processori sono connessi tra loro e condividono la memoria. Ciascun processore ha la stessa priorità nell'accesso alla memoria, può eseguire istruzioni di lettura e scrittura indipendentemente, ed è controllato da un singolo sistema operativo.

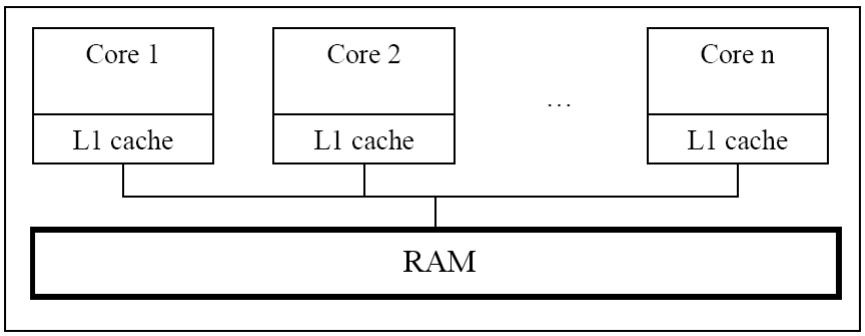

Figure 6.1: Architettura SMP

b) Cluster

Un cluster si realizza quando due o più computer lavorano assieme come fossero un singolo computer: tipicamente sono connessi tramite LAN, e forniscono un'alta disponibilità ai servizi dividendosi il carico di lavoro (si comincia a parlare di "parallelizzazione distribuita").

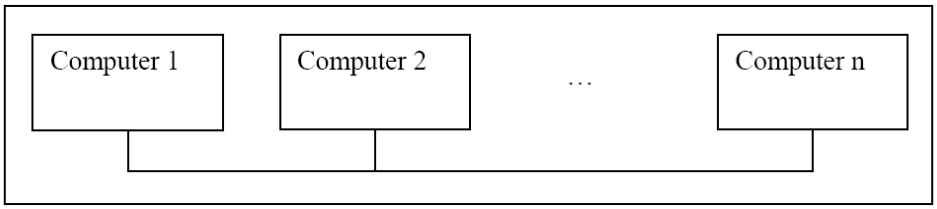

Figure 6.2: Architettura Cluster

c) Ibrida

Quando un cluster è formato da computer di tipo SMP si parla, allora, di architettura ibrida: poiché attualmente la maggior parte dei PC è di tipo SMP, tale architettura risulta essere la più usata in ambito distribuito.

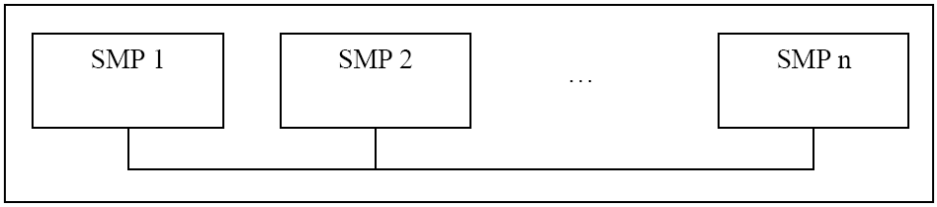

Figure 6.3: Architettura Ibrida

Ad oggi Java fornisce un supporto di base alla parallelizzazione, permettendo di creare thread e processi figli. La classe di base per i thread è java.lang.Threads che esegue un oggetto di tipo java.lang.Runnable. Runnable è l'interfaccia che espone solamente il metodo run() - chiamato da Thread – che andrà a contenere il codice da eseguire in thread separati.

Questi thread possono essere gestiti da un pool, ossia una collezione di thread runnable (in coda) e running creato attraverso la classe statica Executors. Nel pool ciascun thread viene sempre eseguito e controlla nella coda lo stato che dovrà assumere: se vi è la presenza di thread in attesa, allora il pool provvederà ad eseguirli [10].

6.2 Assiomi di partenza

Ecco di seguito alcuni assiomi presi come punto di partenza di questo lavoro di tesi, per delimitare accuratamente l'ambito di lavoro e le conseguenti soluzioni proposte.

L'algoritmo è stato pensato per essere eseguito su comuni PC multicore, che realizzano quindi l'architettura di tipo SMP. Siano:

$M =$ memoria totale disponibile
$P =$ processori
$D =$ dimensione dataset globale

Seguendo la strada del partizionamento parallelo, si ha che ad ognuno dei P processori viene assegnato un proprio dataset D'

$$\text{tale che}$$
$$P * (D' + C) \approx M$$

dove la costante C è rappresenta le eventuali geometrie aggiuntive necessarie al tile in questione.

Si andrà, quindi, a caricare un numero di tile proporzionali alla memoria disponibile, senza sovraccaricarla.

Per quanto riguarda la dimensione del DB e la velocità di trasmissione dei dati, invece, non è stato posto alcun vincolo. Ciò che è noto è il comportamento del DB in fase di lettura/scrittura dati:

1. è possibile effettuare operazioni multiple e simultanee di lettura;
2. è possibile effettuare una singola operazione di scrittura per volta, che richiede l'accesso esclusivo al determinato dataset [1].

[1] Oracle Database Concepts 10g Release 1 (10.1) Part Number B10743-01;

Chapter 7

Conclusioni

In questo lavoro di tesi si è cercato di trovare una metodologia comune per portare a termine la generalizzazione cartografica di differenti classi dopo averne partizionato il dataset iniziale.

L'assunto di base è stato quello di estendere ciascun tile prima di eseguire un qualsiasi algoritmo di generalizzazione, andando poi a salvare solamente tutte quelle geometrie interne al tile di partenza. In questo modo viene garantita la correttezza di ciascuna partizione, nonostante venga inserito un piccolo overhead causato dalla ripetizione dell'esecuzione dell'algoritmo sulle varie situazioni ambigue presenti in prossimità dei bordi. L'estensione, a differenza degli attuali lavori che si trovano in letteratura, viene eseguita in maniera oculata, permettendo all'algoritmo di richiedere le informazioni aggiuntive in maniera dinamica secondo le sue particolari esigenze (senza quindi caricare in memoria, indistintamente, ogni geometria esterna vicina al bordo). Nel caso dei grafi, ad esempio, il MSIG (Minimo Set di Informazione per la Generalizzazione) permette la risoluzione precisa del problema dei grafi, integrando nel dataset solo le informazioni strettamente necessarie.

Si sarebbe potuta prevedere una gestione differente dell'integrazione delle geometrie esterne, per evitare che si venissero a creare ambiguità: una possibilità, ad esempio, poteva essere quella di salvare in modo permanente anche le geometrie sul bordo, facendo sì che non venissero prese in considerazione dalle altra partizioni in cui erano presenti. Si è comunque preferita la gestione delle ambiguità poiché si è ritenuto più importante mantenere il più possibile indipendenti le partizioni, avendo la possibilità di essere eseguite in parallelo, o addirittura – in caso di futuri approfondimenti di questo lavoro – su macchine fisicamente diverse.

La gestione delle ambiguità, essendo peculiari di ciascuna classe, viene effettuata dall'algoritmo stesso che le ha generate, in quanto ne conosce le effettive caratteristiche.

I risultati della generalizzazione dei fiumi possono ritenersi in parte soddisfacenti. Avendo un numero di processi minore del numero di partizioni si nota come le performance non siano superiori alla versione normale dell'algoritmo. Tale caso, però, risulta essere quello della generalizzazione di grandi dataset, che ha la necessità di caricare in memoria solamente dei sottoinsiemi del dataset iniziale: il risultato, perciò, deve essere visto in un'ottica differente, in quanto senza il partizionamento risulterebbe impossibile portare a termine la generalizzazione.

Nel caso di dataset di ridotte dimensioni, potendo caricare in memoria l'intera mappa , si è visto come l'algoritmo riesca, invece, ad avere performance superiori alla versione normale grazie alla simultanea esecuzione in parallelo dei processi sulle varie partizioni.

Futuri lavori possibili sono, senza ombra di dubbio, l'implementazione delle rimanenti classi di algoritmi, portando avanti lo studio per rifinire sempre più la gestione delle ambiguità, e di garantire un miglior bilanciamento del carico di lavoro tra le varie partizioni.

Ringraziamenti

Al termine di questo lavoro di tesi, che coincide con la fine di questo mio lungo "viaggio universitario", vorrei ringraziare tutte quelle persone che negli ultimi anni mi sono state vicine.

Il mio primo pensiero va, naturalmente, alla mia famiglia che è sempre stata presente al mio fianco e non ha mai mancato per un solo attimo di fornirmi il suo prezioso supporto, anche in quelle situazioni in cui tollerare il mio umore diventava impresa assai ardua.

A tutti gli amici che mi sono vicini: quasi sicuramente non riuscirei ad elencarli tutti, e mi rendo conto possa sembrare un'affermazione generica, ma in queste righe vorrei davvero raggiungerli uno ad uno per ringraziarli di cuore per avermi fatto capire come la vera amicizia sia un dono raro e prezioso, motivo per cui mi sento ancor più orgoglioso di essere una piccola parte della loro vita.

E un "Grazie", inoltre, per non avermi mai abbandonato nonostante le mie "vistose carenze" degli ultimi mesi...

Un grazie, in particolare, va a Valentina: per continuare sempre a spronarmi ad essere una persona migliore, e per tutto ciò che mi sta regalando in questo viaggio che non smetterà mai...

Un doveroso ringraziamento, infine, al prof. S. Congiu e a S. Savino, per avermi accolto in questo importante progetto non facendomi mai mancare consigli, critiche e suggerimenti: lavorare in questo team è stata davvero un'esperienza entusiasmante e stimolante.

E ora...si ri-parte!

Bibliography

[1] J. Aquino. Jts topology suite - developer's guide, October 2003.

[2] S. Aronoff. *Geographic Information Systems: A Management Perspective*. 1989.

[3] S. Saha T.J. Alumbaugh D. Tcheng Bajcy, P. P. Groves. A system for territorial partitioning based on gis raster and vector data. Technical report, February 2003.

[4] P.A. Burrough. *Principles of Geographical Information Systems for Land Resources Assessment*. 1986.

[5] W. A. Mackaness Chaudhry O. Partitioning techniques to make manageable the generalisation of national spatial datasets. In *ICA Workshop*, 2008.

[6] T.K. Douglas D.H., Peucker. Algorithms for the reduction of the number of points required to represent a digitised line or its caricature. In *The Canadian Cartographer 10*, volume 2, pages 112–122. 1973.

[7] M. Fondelli. *Italia - Atlante dei Tipi Geografici*, chapter Cartografia tecnica regionale, pages 62–66. 2004.

[8] S. Franceschi. Un introduzione ai gis – sistemi informativi territoriali, 2010.

[9] David Harel and Yehuda Koren. Clustering spatial data using random walks. In *Proceedings of the seventh ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 281–286. ACM, 2001.

[10] Muhammad Usman Iftikhar. Java code transformation for parallelization. Master's thesis, Linnaeus University, School of Computer Science, Physics and Mathematics, 2011.

[11] Anthony K. H. Tung Jiawei Han, Micheline Kamber. Spatial clustering methods in data mining: A survey. In Taylor and Francis, editors, *Geographic Data Mining and Knowledge Discovery*, page 1–29. 2001.

[12] J. S. Keates. *Cartographic design and production, 2nd ed.* copublished in the US with John Wiley & Sons, Inc. New York, 1988.

[13] E. Punt M. Briat, J. Monnot. Scalability of contextual generalization processing using partitioning and parallelization. In *14th Workshop of the ICA commission on Generalisation and Multiple Representation*, Paris, 2011.

[14] D. M. Mark. *The use of quadtrees in geographic information systems and spatial data handling*, volume 1. 1986.

[15] K.S. McMaster, R.B. & Shea. *Generalization in Digital Cartography*. 1992.

[16] S. Mustière and B. Moulin. What is spatial context in cartographic generalisation? In *Geospatial theory, processing and applications*, volume 34, pages 274–278, Ottawa, Canada, 2002.

[17] E. Nicastro. Generalizzazione cartografica della rete stradale nel progetto cargen: algoritmi di armonizzazione, selezione e tipificazione alla scala 1:50.000. Master's thesis, UNIVERSITÀ DEGLI STUDI DI PADOVA, 2011.

[18] Hanan Samet. The design and analysis of spatial data structures. 1990.

[19] E. Schechter. *Handbook of Analysis and Its Foundations*. 1997.

[20] van Harmelen F. Mustière S. Sester M., van Oosterom P. Summary report of dagsthul seminar 09161 on generalization of spatial information. Dagstuhl (Germany), 14-17 April 2009.

[21] M. Trevisani. Cenni sui sistemi informativi territoriali con appunti di geodesia, topografia e cartografia, 04 2003.

[22] C. Huang H. Wu X. Tu, J. Chen. A visual multi-scale spatial clustering method based on graph partition. *INTERNATIONAL GEOSCIENCE AND REMOTE SENSING SYMPOSIUM*, 2:745–748, 2005.

[23] Z. Wu and R. Leahy. An optimal graph theoretic approach to data clustering: theory and its application to image segmentation. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 15(11):1101–1113, 2002.

[24] Chen J. & Zhao R. Zhao, X. Dynamic spatial indexing model based on voronoi. In Science Press, editor, *Proceedings of the International Symposium on Digital Earth*, 1999.

[25] N. Zingarelli. *Lo Zingarelli 2000 : vocabolario della lingua italiana.* 2000.