

EKSPERIMENTALNA PROVERA ALGORITAMA ZA IMPLEMENTACIJU OPERACIJE SPOJA U DISTRIBUIRANOM I HETEROGENOM OKRUŽENJU

Aleksandar Stanimirović, Slobodanka Đorđević-Kajan, Leonid Stoimenov
Elektronski fakultet u Nišu.

Sadržaj – *Pristup geoinformacijama u heterogenom i distribuiranom okruženju, kakvo je okruženje lokalne samouprave, se može razmatrati u kontekstu pristupa podacima u heterogenoj i distribuiranoj bazi podataka. Problem integracije geoinformacija u distribuiranom okruženju se može svesti na problem rekonstrukcije distribuirane tabele sa vertikalnom fragmentacijom. Rekonstrukcija ovakve tabele se vrši primenom operacije spoja nad njenim fragmentima. Rezultati eksperimentalne evaluacija algoritama za implementaciju operacije spoja je pokazala da najbolje performanse ima algoritam indexed nested loops sa primenom rehashing tehnike. Za izbor optimalnog algoritma potrebno je uzeti u obzir i probleme minimizacije saobraćaja na mreži i upotrebe memorijskih resursa.*

1. UVOD

U poslednjih desetak godina razvoj računara i računarskih mreža, posebno Interneta, doveo je do drastičnog povećanja količine informacija koja je dostupna korisnicima informacionih tehnologija. Razmena informacija je postala jedna od osnovni aktivnosti na kojima se zasniva savremeno društvo. Veliki broj organizacija svoje funkcionisanje u velikoj meri zasniva na informacijama koje se neprestano pribavljaju ili razmenjuju sa drugim organizacijama. Zbog same prirode Interneta, koji je globalno neorganizovan, osnovna karakteristika raspoloživih izvora informacija je distribuiranost i heterogenost. Kao posledica toga javljaju se brojni problemi vezani za pronalaženje, identifikaciju i obradu neophodnih informacija.

Geografski informacioni sistemi (GIS) su računarski sistemi čija je osnovna namena upravljanje podacima o prostornim objektima. Nagli razvoj Interneta je doveo i do drastičnih promena u razvoju GIS aplikacija. Umesto monolitnih GIS aplikacija koje rade sa centralizovanim izvorom geoinformacija sve se više javlja potreba za distribuiranim GIS aplikacijama koje će biti u stanju da pribave, obrade i integrišu podatke iz velikog broja distribuiranih i heterogenih izvora geoinformacija (GI). Takođe je potrebno da GIS aplikacije imaju mogućnost saradnje sa drugim sistemima u cilju razmene i obogaćivanja sopstvenih GI. Podaci kojima GIS aplikacije raspolažu moraju biti dostupni što većem broju korisnika kako bi došao do izražaja njihov značaj.

GIS interoperabilnost je jedno od mogućih rešenja problema koji su postavljeni pred savremene GIS aplikacije. Interoperabilnost informacionih sistema se bazira na dogovoru kojim se utvrđuje koji podaci se razmenjuju između izvora GI. Interoperabilnost se bazira i na sposobnosti GIS sistema da slobodno razmenjuju GI bez obzira na formate u kojima se čuvaju ti podaci [1].

Open Geospatial Consortium (OGC) [2] predstavlja organizaciju, koja okuplja veliki broj različitih institucija, čiji je cilj standardizacija procesa razvoja interoperabilnih GIS aplikacija. U skadu sa tim, u proteklih nekoliko godina, OGC je postao vodeća organizacija na polju standardizacije GIS-a. Kao rešenje problema pristupa GI u heterogenom i

distribuiranom okruženju OGC je predložio nekoliko standarda: "OpenGIS Abstract Specification for Features", "OpenGIS Simple Features Specification For SQL", "OpenGIS Simple Features Specification For OLE/COM" i "OpenGIS Simple Features Specification For CORBA".

"OpenGIS Simple Features Specification For OLE/COM" [2] definišu uniformne metode za pristupanje GI bez obzira da li su uskladištene u bazi podataka ili u nekom od internih formata. Ova specifikacija se bazira na Microsoft-ovoj Universal Data Access [3] strategiji za pristup podacima i OLE DB tehnologiji [4]. OLE DB tehnologija obezbeđuje uniformne interfejsa za pristup podacima koji se nalaze u različitim izvorima.

U distribuiranom okruženju, pored problema pristupa izvorima podataka, javlja i problem obrade odnosno integracije pribavljenih podataka. U ovom radu su prikazani eksperimentalni rezultati evaluacije performansi algoritama za integraciju podataka iz većeg broja distribuiranih izvora. Eksperiment i dobijeni rezultati, predstavljeni u ovom radu, se mogu iskoristiti za:

- upoznavanje sa algoritmima za integraciju informacija iz većeg broja distribuiranih izvora podataka i
- procenu i izbor najefikasnijeg algoritma.

U nastavku ovaj rad je organizovan na sledeći način: U drugom poglavlju je opisan problem pristupa prostornim podacima u heterogenom i distribuiranom okruženju na primeru lokalne samouprave. U trećem poglavlju je predstavljeno okruženje za razvoj interoperabilnih GIS aplikacija. Četvrto poglavlje sadrži opis algoritama koji su korišćeni za integraciju GI. Peto poglavlje sadrži detaljan opis samog eksperimenta i uslova pod kojim je sproveden. U šestom poglavlju dati su rezultati eksperimenta i njihovo tumačenje.

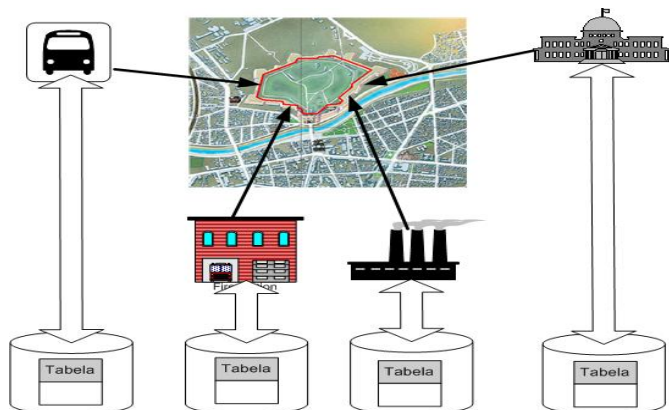
2. PROSTORNI PODACI U DISTRIBUIRANOM I HETEROGENOM OKRUŽENJU

Problem pristupa GI u heterogenom i distribuiranom okruženju biće opisan na primeru lokalne samouprave [5]. Na nivou lokalne samouprave GI generišu različite organizacije (komunalne službe, lokalni Telekom, elektrodistribucija, lokalna policija i ostale službe) koje ili funkcionišu u okviru lokalne samouprave ili su po prirodi svog posla dužne da sarađuju sa lokalnom samoupravom. Broj ovih organizacija se ne može ni na koji način unapred utvrditi ili na bilo koji način ograničiti.

Različite organizacije veoma često dele interesovanje za određenim geobjektom (Sl. 1), ali svaka organizacija ima sopstveni pogled i sopstveno razumevanje objekta od interesa. Kao posledica toga svaka organizacija generiše različite skupove podataka koji opisuju isti geobjekat. Neki od atributa ovih objekata su zajednički za sve organizacije, neki su zajednički samo za neke organizacije, a svaka organizacija može da održava i attribute koji su specifični samo za nju. Neke organizacije koriste GI samo povremeno, dok ih druge koriste veoma intenzivno. U skladu sa svojim potrebama organizacije biraju različite aplikacije i alate za

rad sa GI. To sa druge strane dovodi do toga da se veoma često dešava da različite organizacije u okviru iste lokalne samouprave koriste različite sisteme za skladištenje podataka (rasterske ili vektorske, UNICODE ili non-UNICODE datoteke, strukturne ili nestrukturne dokumente).

U lokalnoj samoupravi GI koje obezbeđuje GIS često mogu da imaju ključnu ulogu prilikom donošenja odluka [5] koje se odnose kako na svakodnevno funkcionisanje lokalne samouprave, tako i na planiranje kratkoročnog i dugoročnog razvoja lokalne zajednice. U samom procesu donošenja odluke često je potrebno konsultovati GI iz različitih izvora. Samim tim potrebno je razviti tehnologije za pristup podacima u distribuiranim i heterogenim izvorima geoinformacija.



Sl. 1 Okruženje lokalne samouprave

Problem pristupa GI u okruženju lokalne samouprave može se posmatrati u kontekstu pristupa podacima u heterogenoj i distribuiranoj bazi podataka. Za sam pristup GI moguće je iskoristiti pomenutu Microsoft OLE DB tehnologiju koja obezbeđuje uniforman tabelarni interfejs za pristup podacima. Integracija tako pribavljenih informacija je nešto složeniji problem.

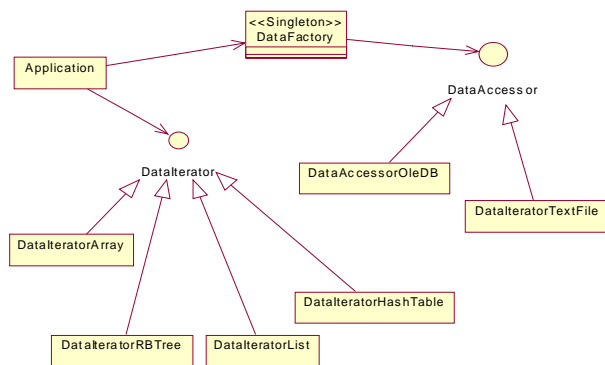
Pretpostavimo da se GI čuvaju kao slogovi jedinstvene tabele geobjekata (GIS *feature* tabele). Svaki geobjekat, odnosno svaki slog u ovoj tabeli, ima svoj jedinstveni identifikator – feature ID (FID). U distribuiranom okruženju je moguće izvršiti vertikalno particionisanje ove tabele na veći broj fragmenata [6]. Svaki fragment obavezno sadrži FID i još neke kolone originalne tabele. Ovakva organizacija geoinformacija odgovara situaciji koja je prikazana na Sl. 1. Svaka od organizacija u okviru lokalne samouprave sadrži odgovarajući vertikalni fragment originalne GIS *feature* tabele (sa onim kolonama koje su od interesa za tu organizaciju). Zahvaljujući postojanju FID-a u svakom od fragmenata moguće je rekonstruisati originalnu tabelu geobjekata. U tu svrhu se koristi JOIN operacija koja se primenjuje nad svim vertikalnim fragmentima tabele geobjekata pri čemu se FID koristi kao uslov spoja. U nastavku rada je prikazana eksperimentalna evaluacija algoritama za implementaciju ove JOIN operacije.

3. RAZVOJ INTEROPERABILNIH GIS APLIKACIJA

U Laboratoriji za računarsku grafiku i GIS na Elektronskom fakultetu u Nišu, razvijen je komponentni okvir pod nazivom GinisFrame [7]. GinisFrame obezbeđuje

veću fleksibilnost i efikasnost u razvoju GIS aplikacija za desktop ili Internet okruženja. GIS aplikacije se razvijaju na bazi ovog okvira kroz razvoj softverskih komponenti koje koriste funkcionalnost komponenti okvira i pristupaju im putem standardnih interfejsa. Postojanje standardnih interfejsa obezbeđuje nezavisnost GIS aplikacija od konkretne implementacije komponente odnosno promena implementacije jedne komponente ne remeti funkcionisanje okvira.

Kao jedna od komponenti ovog okvira razvijen je Ginis OLE DB dobavljač podataka [8]. Ova komponenta obezbeđuje, ostalim komponentama iz okvira, pristup heterogenim izvorima geoinformacija. Komponenta je razvijena na bazi OGC specifikacije “OpenGIS Simple Features Specification For OLE/COM”. Za implementaciju pristupa heterogenim i distribuiranim izvorima geoinformacija ova komponenta koristi DataAccessor projektni obrazac [9] čija je struktura prikazana na Sl. 2. Za integraciju tako pribavljenih podataka koriste se algoritmi koji su opisani u nastavku.



Sl. 2 Projektni obrazac za pristup heterogenim izvorima podataka

4. ALGORITMI ZA IMPLEMENTACIJU JOIN OPERACIJE

JOIN operacija između relacije R i S se definiše kao relacija koja obuhvata sve torke Dekartovog proizvoda (RxS) te dve relacije koje zadovoljavaju specificirani predikat [6]. Predikat je u formi $R.a \theta S.b$, gde je θ bilo koji od relacionih operatora (<, >, =, ...).

Operacija JOIN je izuzetno zahtevna operacija kako po pitanju vremena tako i po pitanju računarskih resursa koje je potrebno angažovati za njeno izvršenje. Zbog toga je za potrebe implementacije ove operacije razvijen veliki broj algoritama čiji je cilj poboljšanje njenih performansi. U ovom radu su razmatrani sledeći algoritmi za implementaciju JOIN operacije [6]:

- *block nested loops* – najjednostavniji algoritam za integraciju gde se spajanje relacija vrši njihovim sekvencijalnim pretraživanjem i upoređivanjem.
- *indexed nested loops* – modifikacija prethodnog algoritma gde se za pretraživanje relacija koristi indeks ili hash funkcija.
- *sort-merge* – vrši spajanje relacija koje su prethodno sortirane po atributima po kojima se vrši spajanje.

5. EKSPERIMENT

Cilj eksperimenta je evaluacija performansi algoritama za implementaciju JOIN operacije u distribuiranom okruženju. Evaluacija je izvršena korišćenjem implementacije ovih algoritama koja je razvijena za potrebe Ginis OLE DB dobavljača podataka. Eksperiment je izvršen na računaru sa Athlon 2500+ procesorom, 1 GB RAM-a i instaliranim Windows 2000 operativnim sistemom.

Algoritmi su proveravani u situaciji u kojoj je potrebno rekonstruisati tabelu geobjekata na osnovu većeg broja distribuiranih fragmenata. Za potrebe ovog eksperimenta kreirana je distribuirana tabela koja se sastoji od tri vertikalno particionisana fragmenta. Fragmenti su smešteni u tri različite Microsoft Access baze podataka. Radi pojednostavljenja ove tri baze su se nalazile na istoj mašini gde se izvršavao i proces koji je obavljao rekonstrukciju tabele geobjekata. Bez ikakvih izmena postupak je mogao da se primeni i za slučaj da su baze podataka bile smeštene na udaljenim računarima i da se podacima moralo pristupati kroz mrežu. Takođe, razmatrana je samo situacija u kojoj veze između fragmenata imaju kardinalnost 1:1, odnosno svaki slog u jednom fragmentu ima odgovarajuće slogove (slogove sa istim FID-om) u ostala dva fragmenta. Isti postupak je mogao biti primenjen i na slučajeve kardinalnosti veza 1:N ili M:N.

U opštem slučaju, u distribuiranom okruženju, svi izvori GI su međusobno nezavisni. To znači da sistem koji vrši

ATL biblioteke [10]: CATlArray za niz, CATlMap za rasutu (hash) tablicu, CATlList za lančanu listu i CATlRbMap za binarno stablo po Red-Black algoritmu. Kod *sort-merge* algoritma uticaj na performanse ima i algoritam koji se koristi za sortiranje podataka. Zbog toga su za ovaj algoritam u razmatranje uzeta dva različita načina za sortiranje podataka: *quick-sort* i *heap-sort* (implementiran korišćenjem binarnog stabla). Za slučaj *indexed nested loops* algoritma korišćena je rasuta tablica, pri čemu su performanse proveravane za dve veličine rasute tablice (500 i 1000 pozicija).

6. REZULTATI

U Tabeli 1 su dati rezultati eksperimentalne provere performansi algoritama za implementaciju JOIN operacije. Ono što na prvi pogleda može da se zaključi je da je zavisnost vremena izvršavanja algoritma od broja slogova koji se obrađuju približno linearna. Takođe može se uočiti da algoritam *block nested loops* ima najgore performanse i rezultati su mu nekoliko desetina puta lošiji od prvog narednog razmatranog algoritma. Algoritam *sort-merge* daje znatno bolje rezultate. Ako uporedimo dve primenjene tehnike sortiranja podataka možemo da uočimo da su razlike veoma male i da se svode na par procenata.

Ako se uporede rezultati *indexed nested loops* algoritama sa rezultatima *sort-merge* algoritama može se uočiti da su razlika na nivou od nekoliko procenata. Međutim rezultati *indexed nested loop* algoritama imaju jednu zanimljivu

Tabela 1 Eksperimentalni rezultati

Algoritam/Broj slogova	Vreme izvršenja algoritma (u sekundama)									
	10000	20000	30000	40000	50000	60000	70000	80000	90000	100000
Block nested loop	7,318	27,793	61,347	108,192	167,472	239,986	328,390	428,394	542,522	669,673
Merge-sort (heap sort)	0,930	1,594	2,283	2,981	3,668	4,350	5,045	5,744	6,446	7,143
Merge-sort (quick sort)	0,943	1,630	2,351	3,092	3,824	4,562	5,322	6,082	6,915	7,801
Index nested loop (n = 500)	0,824	1,442	2,368	2,990	3,966	5,057	6,294	7,662	9,153	10,765
Index nested loop (n = 1000)	0,998	1,789	2,013	2,703	3,442	4,241	5,075	6,060	7,094	8,192
Index nested loop (rehashing)	0,810	1,322	1,852	2,389	2,939	3,418	3,928	4,454	4,976	5,486

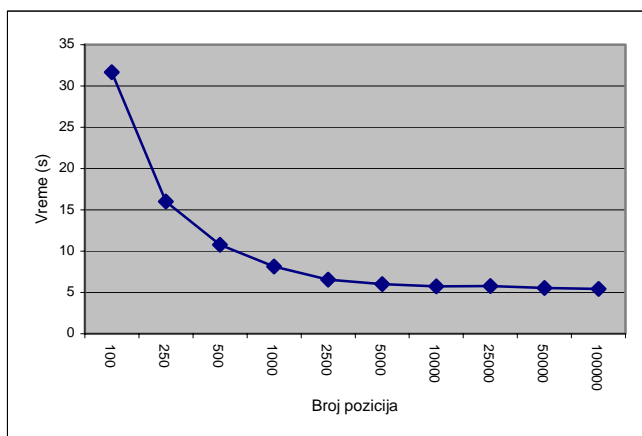
rekonstrukciju distribuirane tabele nema nikakve informacije o broju slogova u pojedinim fragmentima. Zbog toga je u ovom eksperimentu primenjena tehnika gde se najpre pribavljaju slogovi iz svih fragmenata a zatim se vrši njihova integracija. Sa stanovišta memorijskih resursa ovo je najzahtevniji postupak ali najpogodniji za opisanu situaciju. Prilikom određivanja performansi algoritama mereno je ukupno vreme koje je potrebno da bi se izvršila rekonstrukcija distribuirane tabele. Ovo vreme obuhvata: vreme potrebno da se pribave podaci iz svih fragmenata, vreme potrebno da se kreiraju sve neophodne strukture podataka i vreme neophodno za izvršenje samog algoritma. Performanse algoritama su proveravane za tabele sa različitim brojem slogova. Rezultati su dobijeni kao aritmetička sredina deset uzastopnih merenja.

Sve strukture podataka neophodne za realizaciju algoritama su implementirane korišćenjem klasa iz Microsoft

tendenciju. Za manji broj obrađenih slogova ovi algoritmi daju bolje rezultate od *sort-merge* algoritama. Ali povećanjem broja slogova razlika je sve manja, do granice kada *sort-merge* algoritmi počinju da daju bolje rezultate. Ako uporedimo rezultate *indexed nested loops* algoritama sa različitim brojem pozicija videćemo da algoritam sa n=1000 pozicija daje bolje rezultate u odnosu na slučaj sa n=500 pozicija, osim za slučajeve kada se obrađuje mali broj slogova. To je posledica prirode same strukture rasute tablice [10]. Rasuta tablica uz pomoć hash funkcije kreira indeks nad elementima koji se u njoj pamte. Indeks referencira pozicije u rasutoj tablici. Pozicija može da prihvati samo jedan element. Svi ostali elementi koji se preslikavaju na tu poziciju se smeštaju u listu sinonima. Pretraživanje liste sinonima je znatno sporije (jer se vrši sekvencijalno) u odnosu na direktno pristupanje elementu na njegovoj matičnoj poziciji. Zbog toga je potrebno pažljivo izbalansirati broj pozicija u rasutoj tablici u odnosu na ukupan broj elemenata koji se

čuvaju u rasutoj tablici. Što je taj odnos manji to su veće liste sinonima a samim tim se povećava i prosečno vreme traženja u tablici.

Na Sl. 3 su prikazane performanse *indexed nested loops* algoritma za fiksni broj slogova (100000) i različiti broj pozicija u rasutoj tablici. Može se uočiti da u početku povećanje broja pozicija u rasutoj tablici dovodi do drastičnog povećanja performansi algoritma. Sa daljim povećanjem broja pozicija u rasutoj tablici dostiže se određeni optimum pa dalje povećanje više nema značajniji uticaja na performanse. Za implementaciju rasute tablice, kao što je već napomenuto, je iskorišćena Microsoft-ova ATL klasa *CAtlMap*. U dokumentaciji [10] se može naći preporuka da je optimalan odnos između broja pozicija i broja elemenata 0,75. Osim toga preporučuje se i rehashing tehnika. Ova tehnika podrazumeva da se u toku samog procesa kreiranja rasute tablice vrši njeno restrukturiranje, tako da se nastoji da se očuva optimalna vrednost odnosa broja pozicija i broja elemenata. U Tabeli 1 su date performanse *indexed nested loop join* algoritma gde se koriste rasute tablice koje primenjuju rehashing tehniku. Može se uočiti da ovaj pristup daje najbolje rezultate u odnosu na sve ostale algoritme (u nekim situacijama postiže i 25% bolje performanse).



Sl. 3 Performanse *indexed nested loop join* algoritma za različite veličine rasute tablice

ZAKLJUČAK

Osnovni cilj ovog rada je da se na osnovu eksperimentalnih rezultata pronađe najefikasniji način za integraciju rezultata u distribuiranom i heterogenom okruženju kakvo je okruženje lokalne samouprave. Kao što je u radu pokazano problem integracije podataka u distribuiranom okruženju se može svesti na problem rekonstrukcije distribuirane tabele sa vertikalnom fragmentacijom. Rekonstrukcija ovakve tabele se vrši primenom JOIN operacije nad njenim fragmentima.

U radu su prezentovani rezultati eksperimentalne evaluacije performansi algoritama za implementaciju JOIN operacije u distribuiranom okruženju. Prilikom evaluacije od interesa je bilo samo vreme potrebno za integraciju podataka iz različitih izvora. Uzimajući u obzir različite parametre koji

utiču na performanse razmatranih algoritama, može se zaključiti da najbolje performanse postiže algoritam *indexed nested loops* sa korišćenjem rehashing tehnike.

Kao što je već napomenuto od interesa je bilo samo vreme izvršenja algoritama. Ukoliko bi se u sistemu održavala statistika distribuiranih izvora bilo bi moguće izvršiti i evaluaciju razmatranih algoritama sa stanovišta minimizacije saobraćaja na mreži i angažovanja memorijskih resursa. Tek u tom slučaju bi se mogao izabrati optimalan algoritam za integraciju podataka u distribuiranom okruženju.

LITERATURA

- [1] Buehler R., and McKee L., *The Open GIS Guide, Third Edition*, OpenGIS Consortium, Inc, <http://www.OpenGIS.org/techno/specs.htm>, 1998
- [2] Open Geospatial Consortium, <http://www.opengeospatial.org/>, 2005
- [3] Rauch S., *Manage Data from Myriad Sources with the Universal Data Access Interfaces*, Microsoft System Journal, September 1997
- [4] Microsoft Press, *Microsoft OLE DB 2.0 Programmer's Reference and Data Access SDK*, 1998
- [5] Stoimenov L., Djordjević-Kajan S., "Realization of GIS Semantic Interoperability in Local Community Environment", *Proceedings of the 6th Agile Conference on Geographic Information Science*, Lyon, France, April 2003, pp. 73-84
- [6] Elmasri E., Navathe S., *Fundamental of Database Systems*, 4th edition, ISBN 0-321-1222-6, Addison-Wesley, 2002
- [7] Stoimenov L., Stanimirović A., Đorđević-Kajan S., "Realization of Component-Based GIS Application Framework", *7th Agile Conference on Geographic Information Science*, Heraklion, Crete, 29 April-1 May, 2004, pp. 11
- [8] Stanimirović A., Đorđević-Kajan S., Stoimenov L., "Ginis OLE DB dobavljač podataka", *ETAN 2004*, Čačak, Yugoslavia, 2004, pp III.78 – III.81
- [9] Stanimirović A., Đorđević-Kajan S., Stoimenov L., "Primena Data Accessor projektnog obrasca za implementaciju GIS OLE DB dobavljača podataka", *YUINFO 2004*, Kopaonik, 2004, CD izdanje
- [10] Microsoft Corporation, *ATL Library Reference*, <http://msdn.microsoft.com>, 2003

Abstract – Access to geoinformation in heterogeneous and distributed environment, such as local community environment, often can be seen in context of heterogeneous and distributed databases. Problems of geoinformation integration in distributed environment can be treated as problem of recreating distributed table with vertical fragmentation. These kinds of tables are reconstructed using join operation on vertical fragments. Experimental evaluation of join algorithms has shown that *indexed nested loops* algorithms with rehashing techniques have best performance. In order to choose optimal algorithm problems of minimal network traffic and memory resources also have to be evaluated.

EXPERIMENTAL EVALUATION OF JOIN ALGORITHMS IN DISTRIBUTED AND HETEROGENEOUS ENVIRONMENT

Aleksandar Stanimirović, Slobodanka Đorđević-Kajan, Leonid Stoimenov