타일-기반 수치지도를 이용한 동시 합병 처리기 설계 및 구현

Design and Implementation of Concurrent Merge System Using Tile-based Digital Maps

이상현 *, 김동현**, 홍봉희***
Sang-Hyun Lee, Dong-Hyun Kim, Bong-Hee Hong

요 약 기존에 제작되어 있는 수치지도를 이용해서 공간 데이타베이스를 구축할 때 타일 기반의 분리된 수치지도를 Seamless Map으로 다시 제작하는 과정은 필수적이다. 기존의 반복적인 합병 방법은 작업의 동시성이 떨어지는 문제점이 있다. 이 논문에서는 다수 작업자에 의한 작업의 동시성을 유지할 수 있도록 하는 변형된 2PC 프로토콜에 기반한 동시 합병 처리기의 구현에 대해 다루고 있다. 작업자들은 처리기를 통하여 서로 간의 작업을 공유할 수 있으므로 합의에 의한 합병 작업을 동시에 진행할 수 있다.

ABSTRACT To build a spatial database from digital maps, it is necessary to rebuild a seamless-map with discrete tile-based digital maps. The sequential merge methods have the disadvantage of degenerating concurrency. In this paper, we consider the implementation of the system based on modified 2PC protocol to retain the concurrent merge. As the engineers share the merge through the system, they can concurrently proceed to merge by negotiation.

키워드 : 수치지도, 합병, 동시 합병, 협동작업, Seamless Map

1. 서 론

GIS(Geographic Information System)에서 공간 데이타베이스의 구축은 많은 시간과 노력을 필요로 하지만, 수치지도(Digital Map)의 제작이 활성화됨에 따라 수치지도를 이용한 공간 데이타베이스 구축이 주로 사용되고 있다. 공간 데이타를 획득하는 과정에서 측량 과정이 제외되므로 비용을 절감할 수 있기 때문이다. 수치지도를 제작하기 위해 사용되는 방법으로는 기존의 종이 지도(Paper Map)를 디지타이저를 이용해 수동으로 입력하는 방법, 스캐너 등에 의해 디지털 이미지로 변환한 후 선형화 하는 방법, 항공사진측량(Photogrammetry) 등을 이용하는 방법 등과 같이 다양한 방법이 있지만, 그 중에서 항공사진측량에 의한 방법이 자료의 정확도, 경제성, 자료의 갱신에 따르는 비용 등을 고려할 때에 가장 효과적인 방법으로 알려져 있다.[19] 항공사진측량을 이용한 수치지도 제작 과정을 보면, 우선 제작하고자 하는 대상 지역에 대한 촬영 계획을 수립하고 지상 및 사진 기준점 측량을 수행한 후 해석도화(digital mapping) 과정을 거쳐서 1차 수치지도를 완성한다. 이렇게 제작된 수치지도는 현지 조사와 항측 보완 측량 그리고 마지막으로 정위치 편집을 함으로써 최종 완성된다.[20][21] 수치지도를 제작할 때는 일반적으로 타일(Tile)이라고 부르는 작은 조각(piece)으로 나누어서 작성한다. 넓은 지역에 해당하는 자료를 컴퓨터에 입력하여 관리할 때 관리 목적상 작은 단위 면적으로 나누어 관리하는 것이 여러모로 편리하고, 또한 수치지도 데이타 파일

† 연구세부분야 : 데이타베이스(GIS DB)

* 부산대학교 지형정보협동과정
** 부산대학교 컴퓨터공학과
*** 부산대학교 컴퓨터공학과 교수
Email: {lsh12, pusrover, bhhong}@hyowon.cc.pusan.ac.kr

에 대한 효율적인 검색기능을 제공함으로써 시스템의 제반 성능을 향상시킬 수 있기 때문이다.[20] 결과적으로 최종 완성된 수치지도는 많은 수의 타일로 구성된다.

공간 데이타베이스 구축을 위해 타일-기반의 수치지도를 이용하는 경우에는 먼저 Seamless-Map 제작을 위해 수치지도 간의 합병(merge) 작업이 선행되어야 한다[5]. 수치지도별로 공간 데이타베이스를 구축하는 경우에는 각 수치지도에 대한 DB구축비용 이외의 부가적인 비용이 들지 않으므로 구축 비용이 저렴하다. 그러나 동일한 속성을 가진 객체(object)가 여러 개의 서브 객체(sub-object)로 분리 저장되기 때문에 공간 분석을 수행할 때 Join등과 같은 공간 연산이 부가적으로 수행되어야 하는 등 분석 알고리즘이 복잡해지고, 동일한 속성(attribute)의 데이타를 다수의 튜플(tuple) 혹은 객체에 중복 저장해야 함에 따라 저장공간의 낭비는 물론 데이타베이스의 일관성(Consistency)유지를 위한 추가적인 작업이 필요하게 되는 단점이 있다. 수치지도를 Seamless-Map으로 재작성한 뒤 공간 데이타베이스를 구축한 경우에는 DB 구축비용 이외에 지도 조인(Map Join)을 위한 합병 비용이 추가되므로 구축비용이 증가하는 단점이 있다. 그러나 공간 분석 알고리즘이 상대적으로 단순해지고 저장공간의 절약과 데이타베이스 일관성 유지를 위한 부가적인 작업이 필요 없는 장점이 있다.

Seamless-Map을 제작하기 위한 기존의 방법은 각각의 수치지도에 대해 작업자들이 개별적으로 합병 작업을 수행한 후 합병된 수치지도에 대해 다시 반복적으로 합병하는 Tree형태의 반복 합병 방법이었다. 반복 합병 방법은 작업이 진행됨에 따라 작업에 참여할 수 있는 작업자의 수가 점차 줄어들게 되고 그에 따라서 작업의 생산성이 저하되는 문제점이 있다. 그러므로 Seamless-Map 제작의 작업 생산성 향상을 위해 반복적인 합병 작업 없이 다수의 작업자가 동시에 합병 작업을 수행할 수 있는 처리기에 대한 연구가 필요하다.

이 논문은 다수의 작업자가 동시 합병 작업을 수행하기 위해 그림 1과 같이 클라이언트가 자신의 로컬 사이트에 수치지도의 일부를 캐싱하고 있는 클라이언트-서버 환경을 가정한다.

그림 1의 환경에서 Seamless-Map을 작성할 경우에는 그림 2와 같은 문제점들이 발생한다. 첫째, 작업자가 합병 작업을 수행한 뒤 서버에 저장하는 시간적 선후 관계에 따라 앞선 작업자의 작업을 잃어버리는 작업 손실(Lost-Work)이 발생한다(그림2.(가)). 둘

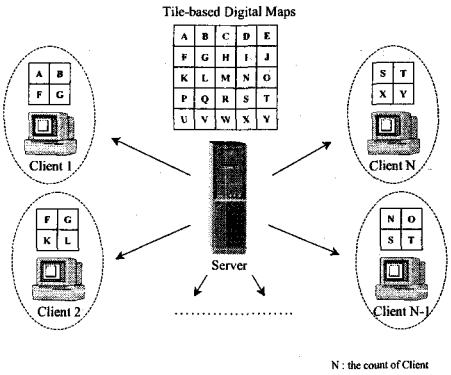


그림 1. 시스템 환경

째, 다수 작업자가 같은 엔티티들에 대해 서로 교차해서 잠금을 요청하는 경우에 교착상태(Deadlock)가 발생한다(그림2.(나)). 셋째, 수치지도 데이타가 각 클라이언트에 중복 저장되어 있음으로 인해 작업이 진행됨에 따라 전체 수치지도 들의 일관성이 유지되지 못하는 문제가 발생한다(그림2.(다)).

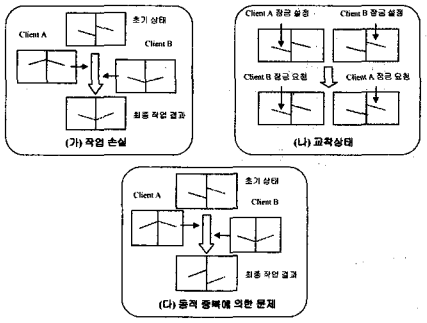


그림 2. 동시 합병 작업의 문제점

이 논문에서는 클라이언트-서버 환경 하에서 수치지도에 대한 동시 합병 작업시 발생하는 이러한 문제점들을 해결하기 위해 작업자 상호간의 메시지 교환에 의한 합병 작업을 통하여 서로의 작업을 공유함으로써 작업의 동시성(Concurrency)을 확보할 수 있는 동시 합병 작업 프로토콜을 제안하고 이 프로토콜에 따른 세부 알고리즘을 제시한다. 또한, 합병 작업에 의해 변경된 엔티티를 중복 저장하고 있는 클라이언트에 대한 변경 전파에 대해 기술하고 해당 알고리즘을 제시한다. 마지막으로 제시된 알고리즘들을 이용한 수치지도 동시 합병 처리기를 설계 및 구현한다.

이 논문의 구성은 다음과 같다. 2장에서는 이 논문

과 관련된 연구들을 기술하고, 3장에서는 동시 합병 작업에 대하여 기술하며, 프로토콜 및 알고리즘을 4장에서 기술하고, 5장에서는 수치지도 동시 합병 처리기를 위한 시스템의 설계 및 구현 화면을 보이고 마지막으로 6장에서 결론 및 향후 연구 과제에 대해 기술한다.

2. 관련 연구

기존의 제작되어 있는 타일-기반 수치지도를 이용해서 공간 데이타베이스 구축을 할 수 있는 상용 프로그램은 전세계의 많은 업체들에 의해 이미 개발, 보급되어 실제 산업현장에서 널리 사용되고 있다. Arc/Info(ESRI사, 美), Gothic(LaserScan사, 英), MapInfo(MapInfo사, 美) 등이 국내외에 널리 알려져 있는 대표적인 제품들이다. 이외에도 많은 상용 제품들이 있지만 이러한 기존의 상용 제품들은 수치지도에 대한 다수 작업자의 동시 합병 작업을 지원하지 않기 때문에 앞서 기술한 반복 합병 방법을 사용하므로 작업의 생산성이 떨어지는 단점이 있다.

[2]에서는 클라이언트-서버 환경에서 클라이언트 캐쉬의 일관성 유지와 변경 전파에 대한 알고리즘을 제안하고 있다. 특히, 다수의 클라이언트가 중복된 영역에 대한 수정 작업을 할 경우의 트랜잭션 모델을 위해 [9]에서 정의한 영역 잠금을 기반으로 한 CS(Cached Shared)-잠금, CX(Cached eXclusive)-잠금 등의 확장된 잠금 기법을 제안하고 있다. 그러나, 작업공간 간의 경계선에서 만나는 엔티티들에 대해서 합병 작업을 하고자 할 경우에 영역 잠금 방식으로는 불필요한 영역까지 잠금을 설정해야 하는 문제가 발생하기 때문에 [2]에서 제시한 트랜잭션 모델을 적용할 수가 없다. 예를 들면, 도로 중심선에 대한 합병작업을 하고자 할 경우에 이미 다른 클라이언트들에 의해 모두 합병작업을 했다고 가정하면 자신의 영역에 포함된 도로 중심선 및 다른 클라이언트들에 의해 수정된 도로 중심선들을 모두 포함하는 영역 잠금을 설정해야만 하며, 다른 클라이언트의 작업 공간을 모두 포함하는 경우까지도 발생하게 된다. 따라서, 서로 관련성이 없는 클라이언트끼리 불필요한 메시지 통신을 하게 되며 그로 인한 오버헤드가 발생하게 된다.

[5]에서는 긴 트랜잭션일 경우 널리 쓰이고 있는 체크아웃(Check-out) 모델에 관해 기술하고 있다. 이 모델에서는 작업자가 임의의 영역에 대한 객체들을 서버의 데이타베이스로부터 자신의 저장소에 복사한 뒤 작업이 완료된 객체에 대해 서버의 데이타베이스로 체크인(Check-in)하는 방법을 사용한다. 그러나 이 방법은 기본적으로 잠금을 사용하기 때문에 한 사람의 작업자가 체크아웃을 하게 되면 다른 작업자는 체크인이 되기 전까지 긴 시간 대기해야만 하므로 이 논문에서와 같은 동시 합병 트랜잭션을 수행하기에는 적합하지 않다.

[3]에서는 클라이언트마다 독자적인 작업공간을 가지고 독자적인 객체 변경이 가능하며 변경 전의 객체를 중복 저장하고 있는 다른 클라이언트에게 객체의 변경 내용을 전파를 하는 알고리즘으로 "두 단계 델타 합병(two-phase delta-merge) 알고리즘"을 제안하고 있다. 그러나 이 알고리즘은 클라이언트가 객체를 수정할 때마다 변경 전파를 하기 때문에 잦은 객체 수정이 발생할 경우에는 메시지 과부하 현상이 발생하게 된다. 또한 클라이언트는 자신의 작업공간에 포함된 객체에 대해서만 수정 작업이 가능하므로 이 논문에서 다루고자 하는 합병 작업을 위한 알고리즘으로 적용할 수 없다.

3. 동시 합병 작업

3.1 용어 정의

이 논문에서 사용하는 용어들에 대한 정의를 한다.

동시 합병 작업 : 이 논문에서는 타일-기반의 수치지도에 대해 다수의 작업자가 동시에 합병 작업을 수행할 때 발생하는 문제점들을 해결하기 위해 동시 합병 작업 프로토콜을 사용한다. 이 프로토콜에 의해 두 사이트 이상의 클라이언트가 서로 인접한 수치지도의 엔티티들을 동시에 합병하는 작업을 이 논문에서는 동시 합병 작업이라고 정의한다.

• 작업공간(Workspace) : 동시 합병 작업을 위해서 우선 대상 지도 전체를 일정 영역으로 타일링(Tiling)한 뒤 각 타일을 클라이언트들에게 할당한다. 여기에서 클라이언트에게 할당되는 하나 이상의 타일 셋을 이 논문에서는 작업공간(Workspace)이라고 정의하며 하나의 클라이언트는 오직 하나의 작업공간만을 가진다고 가정한다. 또한 작업공간 내에 포함된 엔티티들에 대한 수정 권한은 오로지 그 작업공간에 할당된 클라이언트에게만 주어진다.

• 조정자(Coordinator) : 대리인에게 합병 작업을 요청하는 클라이언트를 조정자라고 정의한다.

• 참여자(Participant) : 대리인으로부터 합병 작업 요청 메시지를 받는 클라이언트를 참여자라고 정의한다.

• 대리인(Agent) : 서버를 말하며 조정자의 합병 작업 요청이 왔을 때 참여자에 대한 검색 및 메시지 전달 역할을 하며 합병 작업이 완료되었을 때 합병 결

과를 저장하고 관련 클라이언트들에게 전파하는 역할을 한다.

• 관련자(The other) : 조정자 및 참여자에 의해 합병 작업이 종료된 후 해당 엔티티를 중복 저장하고 있는 클라이언트를 관련자라고 정의한다.

3.2 잠금 (Locking)

클라이언트의 작업공간은 수치지도 한 매 이상의 집합으로 구성되며 작업공간 내에 포함된 엔티티에 대한 수정 권한은 오직 해당 클라이언트에게 주어진다. 또한 지도에 대한 수정 작업은 그래픽 화면을 보면서 대화식으로 이루어진다는 특성을 가지고 있기 때문에 어떤 클라이언트가 다른 클라이언트의 작업공간에 존재하는 엔티티들에 대한 참조가 필요한 경우 기하 데이타를 읽을 수 있어야 한다. 작업공간에 대해 X 잠금이 설정된 경우 다른 클라이언트들의 참조가 불가능하게 되므로, 이 논문에서는 이러한 제약조건을 해결하고 클라이언트의 작업공간을 정의하기 위하여 영역잠금(Region Lock)[9]을 사용한다. 영역잠금은 잠금의 단위(granularity)를 각 클라이언트의 작업공간에 대해 한정함으로써 작업의 동시성을 높이고자 하는 기법으로서 READ 잠금을 허용하는 weak-SIX (shared intention exclusive)잠금이다. 영역잠금은 기존의 잠금 모드인 SIX 잠금 모드와 유사하지만 다른 클라이언트의 READ를 허용하는 특징이 있다. 영역잠금을 사용함으로써 다른 클라이언트들이 자신의 작업공간이 아닌 다른 작업공간에 존재하는 기하 데이타에 대한 참조가 가능해진다. 또한 이 논문에서는 서버에 저장된 기하 데이타에 대한 X잠금과 S잠금을 사용한다. 이 논문에서 사용하는 잠금 들의 호환성은 표 1과 같다.

표 1. 잠금 호환성 테이블

	Region	X	S
Region	Yes	No	Yes
X	Yes	No	No
S	Yes	No	Yes

3.3 동시 합병 작업 프로토콜

3.3.1 합병 작업의 유형

두 클라이언트의 경계에서 나타나는 합병 작업의 유형은 그림 3과 같이 분류할 수 있다. 이 분류는 합병 작업 시 나타날 수 있는 작업들의 전형을 보여주고 있다. 그림에서 Type 1, Type 2, Type3, Type 7은 관련자가 없는 합병 작업을 보여주고 있으며 나머지 Type은 관련자가 있는 합병 작업을 보여주고 있다.

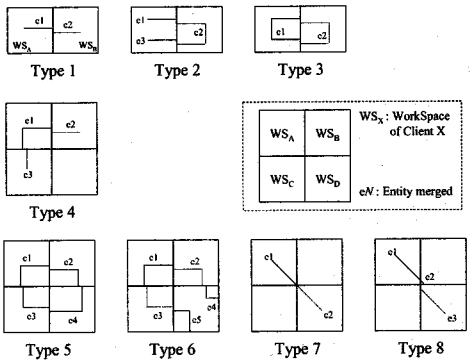


그림 3. 합병 작업의 유형 분류

이러한 분류에 의해 합병 작업을 정리하면 표2와 같다.

표 2. 합병 작업의 유형 분류

Type	Entities	Coordinator (Client X)	Participant (Client X)	The others (Client X)
1	e1, e2	A	B	-
2	e1, e2	A	B	-
	e1, e3	A	B	-
3	e1, e2	A	B	-
4	e1, e2	A	B	C
	e1, e3	A	C	B
5	e1, e2	A	B	C, D
	e1, e3	A	C	B, D
	e2, e4	B	D	A, C
	e3, e4	C	D	A, B
6	e1, e2	A	B	C, D
	e1, e3	A	C	B, D
	e2, e4	B	D	A, ···
	e3, e5	C	D	A, ···
7	e1, e2	A	B	-
8	e1, e2	A	B	D
	e1, e3	A	D	B
	e2, e3	B	D	A

3.3.2 동시 합병 작업 연산

클라이언트/서버 환경에서 동시 합병 작업을 위한 통신 프로토콜을 기술하기 위한 동시 합병 작업 연산을 기술한다. 동시 합병 작업 연산은 전통적인 트랜잭션 연산을 확장하고 합병 작업의 결과를 다른 사이트의 클라이언트들에게 전파하는 연산 등을 추가한 것이다. 동시 합병 작업 연산은 사용자에 의해 실행되는 연산과 시스템에 의해 자동 실행되는 연산으로 구분이 된다.

먼저, 사용자에 의해 실행되는 연산은 다음과 같다.

• Request-Merge : 합병 작업 대상 엔티티에 대해 조정자가 선택을 마친 후 대리인에게 합병 작업을 요청한다. 또는 합병 작업을 요청 받은 대리인이 참여자를 검색한 후 성공하면 검색된 참여자에게 합병 작업을 요청한다.

• Msg_req_merge - 조정자가 엔티티 합병 작업을 대리인에게 요청한다.

- Msg_rep_cancel_lock - 합병 작업 대상 엔티티가 잠금 상태임을 조정자에게 알린다.
- Msg_rep_ok - 합병 작업 요청에 대해 허락을 알린다.
- Msg_rep_cancel - 합병 작업 요청에 대해 거부를 알린다.

• mid-commit : 동시 합병 작업 대상 엔티티에 대한 참여자와의 합의 후 합병 작업이 완료되었을 때 조정자가가 대리인에게 합병 결과에 대한 저장 및 전파를 요구한다.

시스템에 의해 자동 실행되는 연산은 다음과 같다.

• Set-Region-Lock : 클라이언트가 시스템을 구동할 때, 이미 설정되어 있는 작업공간 정보를 이용하여 대리인에게 영역 잠금을 설정을 요청한다.

• Set-X-Lock : 조정자가 대리인에게 합병 작업 요청을 했을 경우 대상 엔티티들 중에서 조정자의 작업공간에 포함되는 엔티티에 대한 X 잠금을 설정한다. 또한 참여자가 합병 작업 허락을 응답했을 경우 참여자의 작업공간에 포함되는 엔티티에 대한 X 잠금을 설정한다.

• Propagate-Update : 합병 작업이 완료된 후 대리인은 참여자 및 관련자들에게 엔티티에 대한 합병 작업이 완료되었음을 통보하여 캐쉬의 일관성을 유지하도록 한다.

3.3.3 동시 합병 작업 프로토콜

조정자는 작업공간의 외곽 경계선에서 라인의 시작 노드 혹은 끝 노드가 서로 만나거나 근접해 있는 엔티티들에 대해 합병 작업을 하고자 할 경우에 먼저 대리인에게 합병 작업 요청 메시지를 보내어 참여자와의 합의에 의해 작업을 공유해야 한다. 그림 4에서는 조정자와 참여자 간의 합병 작업을 위한 프로토콜을 보이고 있다. 이 프로토콜은 기존의 2-PC 프로토콜의 변형된 형태이다. 기존의 2-PC 프로토콜은 작업의 결과에 대해 참여자들에게 완료여부를 확인하는 과정과 완료 내용의 전파를 위한 과정으로 구성되어 있다. 그러나 동시 합병 작업 프로토콜은 작업의 허가를 얻기 위한 과정과 완료 내용의 전파를 위한 과정으로 구성되어 있다.

조정자가 자신의 작업공간에 포함된 엔티티인 a를 참여자의 작업공간에 포함된 엔티티인 b와 합병시키고자할 경우에 조정자는 참여자로부터 엔티티 b에 대한 작업 허가를 얻어야만 작업을 진행할 수 있다. 합병 작업이 진행되는 과정을 단계별로 정리하면 다음과 같다.

1. 조정자가 엔티티 a와 엔티티 b에 대한 합병 작업을 요청하는 메시지(MSG_REQ_MERGE)를 대리인에게 보낸다.
2. 대리인은 작업공간 정보와 엔티티 a, b의 기하정보간의 위상 관계를 이용하여 참여자를 검색한다.
3. 검색된 참여자에게 대리인은 동시 합병 작업을 요청하는 메시지 (MSG_REQ_MERGE)를 보낸다.
4. 참여자가 대리인의 요청을 accept한 경우에는 요청 메시지에 포함된 수정 엔티티에 대한 정보를 이용하여 자신의 캐시를 수정하고 동시에 대리인에게 작업 허가 메시지 (MSG_REP_OK)를 보낸다.
5. 대리인은 MSG_REP_OK 메시지를 조정자에게 전달한다.
6. 대리인은 조정자로부터 MSG_REQ_UPDATE 메시지가 오면 합병된 기하 데이타에 대한 디스크 쓰기 작업을 수행한다.
7. 상기 1 ~ 6까지의 과정을 반복한다.

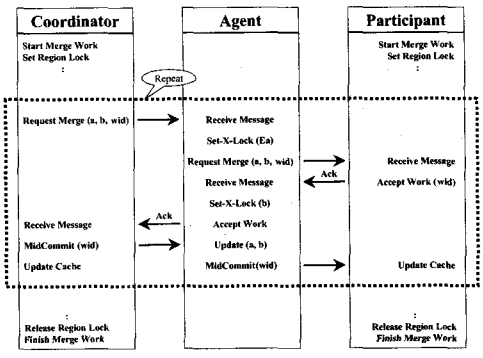


그림 4. 동시 합병 작업 프로토콜

조정자가 대리인에게 합병 작업 요청 메시지를 보낼 때와 대리인이 참여자에게 요청 메시지를 보낼 때에는 기하정보를 포함한 합병 작업에 필요한 기타의 모든 정보를 보내지만 그 외의 경우에는 네트워크 통신 오

버헤드를 줄이기 위해 메시지의 OID를 이용한다.

조정자의 합병 작업 요청이 거부되는 경우는 2가지로 구분할 수 있다. 첫번째는 조정자가 요청한 합병 작업 대상 엔티티가 이미 다른 조정자에 의해 합병 작업이 진행 중인 경우이다. 이 경우에 대리인은 엔티티에 대한 X 잠금을 설정해둔 상태이므로 조정자의 작업 요청을 거부하는 메시지(Msg_rep_cancel_lock)를 보내게 된다. 두 번째는 참여자가 조정자의 작업 요청을 거부하는 경우이다. 이 경우는 참여자가 조정자의 요청을 직접적으로 거부하는 경우도 있겠지만 참여자가 일정 시간 이상 응답이 없을 경우도 포함이 된다.

4. 변경 전파 메커니즘

대리인은 조정자로부터 mid-commit 요청이 왔을 때 이에 대한 처리를 완료한 뒤에 관련자들과의 캐쉬 일관성 유지를 위해 합병 작업 결과 정보를 관련자들에게 전파해야 한다. 이 장에서는 대리인의 변경 전파에 대한 내용을 기술한다.

4.1 작업공간 정보 테이블

대리인은 조정자로부터 합병 작업 요청 또는 mid-commit 요청이 왔을 때 메시지 전파 대상이 되는 참여자 및 관련자들을 검색하기 위해 각각의 클라이언트들에게 할당된 작업공간에 대한 정보를 담고 있는 테이블을 저장한다. 이 논문에서는 이 테이블을 작업 공간 정보 테이블 (Workspace Information Table:이하 WIT)이라고 정의하며 다음과 같은 내용을 담고 있다.

- 클라이언트 ID : 각 클라이언트를 식별하는 식별자
  예) IP Address : 164.125.69.161
- 작업공간 영역 좌표 : 작업공간을 구성하는 좌표 집합
  예) MBR : (minx, miny, maxx, maxy)

대리인은 WIT와 엔티티의 기하좌표를 이용하여 조정자, 참여자 및 관련자들을 검색한다. 검색을 위한 공간 연산자로 'Meet'를 사용하며 검색방법은 먼저 합병하고자 하는 엔티티 또는 합병된 결과 엔티티의 기하좌표로부터 시작 노드와 끝 노드를 검색한 뒤, 각 노드와 'Meet' 관계인 작업공간을 WIT의 작업공간 영역 좌표로부터 검색한다. 참여자는 1개가 검색이 되지만 관련자들은 최소 2개에서 최대 8개까지 검색이 가능하다. 검색된 참여자 및 관련자들에게 대리인은 합병 요청 메시지 혹은 합병 결과 전파 메시지를 보내게 된다.

4.2 변경 전파 메커니즘

그림 3의 Type 4, 5, 6, 8의 경우에 합병 작업이 발생하면 대리인은 관련자들에게 합병된 결과를 전파해야 한다. 조정자로부터 mid-commit 요청이 대리인에게 도착한 시점으로부터 관련자들의 변경 결과 수정까지를 단계별로 정리하면 아래와 같다.

1. 조정자 : 대리인에게 mid-commit 요청
2. 대리인 : WIT를 이용하여 관련자들 검색. 합병 결과 엔티티의 시작 및 끝 노드와 만나는 작업 공간이 관련자가 된다. 만약, 합병 결과 엔티티의 그림 3의 Type 1, 2, 3, 7의 경우에는 합병 된 결과 엔티티의 시작 및 끝 노드가 조정자 및 참여자의 작업공간 내부에 존재하기 때문에 관련자들이 검색되지 않는다. 그러나 Type 4, 5, 6, 8의 경우에는 합병 결과 엔티티의 시작 및 끝 노드 중 한쪽 또는 양쪽 모두가 조정자 및 참여자의 작업공간 이외의 다른 작업공간과 'Meet' 하므로 관련자들이 검색된다.
3. 대리인 : 검색된 관련자들에게 변경 전파 요청 메시지와 함께 합병 작업 전 엔티티의 기하정보와 합병 작업 후 엔티티의 기하정보를 보낸다.
4. 관련자들 : 대리인으로부터 온 합병 작업 전 엔티티의 기하정보를 이용하여 해당 엔티티를 검색한 후 결과 엔티티의 기하정보를 이용하여 변경된 내용을 수정한다.

상기 단계의 프로토콜은 그림 5와 같다.

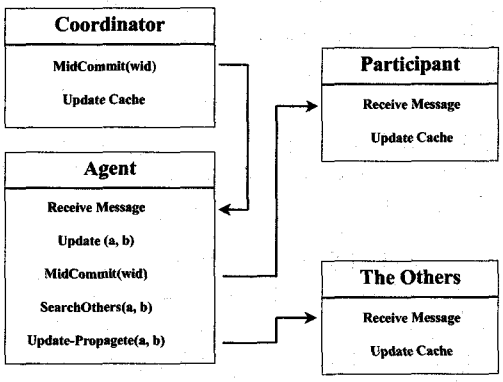


그림 5. 변경 전파 프로토콜

5. 동시 합병 작업 알고리즘

5.1 동시 합병 작업 알고리즘

그림 6의 (가)와 (나)는 각각 클라이언트(조정자, 참여자, 관련자)와 대리인의 경우에 대해 동시 합병

(가) 클라이언트 세부 알고리즘

```
Client CoWork for SendMessage {
        If (Request Merge)          // 서버에게 합병 작업을 요청할 때
          Select Entities by user and Send Request-Message to server
      Else                          // 서버로부터 온 메시지에 대한 응답을 할 때
          Send Reply-Message to server
}
Client CoWork for ReceiveMessage {
        Get Reply from server
        Switch (Reply) {
                Case MSG_REP_OK :                   // 합병 작업이 수락되었을 때
                     Proceed Request Work
                Case MSG_REP_CALCEL :    // 합병 작업이 거부되었을 때
                Case MSG_REP_CANCEL_LOCK :          // 합병 작업이 잠금에 의해 거부되었을 때
                     Cancel Work
                Case MSG_REQ_UPDATE :    // 변경 전파 요청이 왔을 때
                     Update Cache Map
                Case MSG_REQ_MERGE :     // 합병 작업 요청이 왔을 때
                     Do Request Merge UI
        }
}
```

(나) 서버 세부 알고리즘

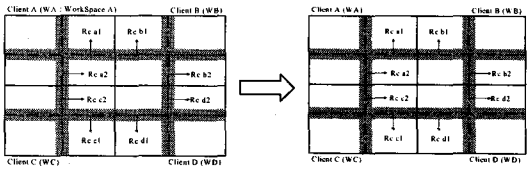
```
Server CoWork for ReceiveMessage {
        Get Message from Client (Request or Related)
        Switch(Message){
                Case MSG_REQ_EDGEMATCH :
                Case MSG_REQ_MERGE :
                        If (CheckLock is TRUE) // 요청 엔티티가 잠금 상태이면
                                Reply MSG_REP_CANCEL_LOCK to Request-Client
                        Else
                                Search Related-Client & Send Message to Related-Client
                Case MSG_REP_OK :
                        Write Data to Log                   // 로그 파일에 기록
                        Search Request-Client
                        Reply Message to Request-Client
                Case MSG_REP_CALCEL :
                        Search Request-Client and Reply Message to Request-Client
                Case MSG_REQ_UPDATE :
                        Update Digital Map
                        Search Prop-Clients and Propagate Update Entities
                        Write Data to Log
        } // End of Case
} // End of ReceiveMessage
```

그림 6. 동시 합병 작업 세부 알고리즘

작업을 위한 메시지 처리 세부 알고리즘을 보이고 있다. 클라이언트는 합병 작업 요청 메시지를 대리인에게 보내는 부분과 조정자에 의해 요청된 합병 작업을 대리인으로부터 받는 두 부분으로 크게 나누어지며, 대리인으로부터 전달되어 오는 메시지의 경우에는 동시 합병을 작업 요청 메시지와 캐시의 일관성 유지를 위한 변경 요청 메시지로 나뉜다. 대리인은 조정자가 가 요청하는 작업에 대해 참여자의 응답을 되돌려 주거나 합병 작업 결과에 따라 관련자들에게 처리 결과를 메시지로 전파한다.

5.2 동시 합병 작업의 예

그림 7은 타일 기반의 수치지도로 구축되어 있는 도로 중심선을 합병하는 예를 보인 것이다. 그림에서 네 명의 작업자가 네 개의 작업공간으로 분할된 대상



(가) 합병 전 (나) 합병 후

그림 7. 도로 중심선 합병의 예

지역에 대해 각 작업공간 사이의 도로 중심선들을 합병하는 작업을 동시에 진행한다. 그림의 예에서 클라이언트 A는 클라이언트 B 혹은 클라이언트 C와 합병 작업을 진행하고, 나머지 클라이언트들도 마찬가지로 인접한 클라이언트와 합병 작업을 동시에 수행한다. 이 경우에는 합병을 하고자 하는 각각의 엔티티들이 모두 분리되어 있기 때문에 하나의 클라이언트가 합병 작업을 진행할 때 다른 클라이언트들이 진행하는 합병 작업에 영향을 끼치지 않는다.

6. 설계 및 구현

이 논문에서는 클라이언트/서버 환경을 기본으로 클라이언트 및 서버의 모든 모듈을 객체지향 언어인 C++로 구현하였다. C++ 컴파일러 및 사용자 인터페이스 구성을 위하여 마이크로소프트사의 Visual C++ 5.0을 사용하였으며 클라이언트의 OS로는 Windows 98, 서버의 OS로는 Windows NT를 사용하였으며 각 시스템들은 네트워크로 연결이 되어 있다. 수치지도는 국립지리원에서 제작한 서울특별시 1/5,000 도엽 수치지도(DXF 파일)를 사용하였다.

6.1 시스템 설계

그림 8에서는 동시 합병 작업 알고리즘을 기반으로 한 시스템의 구성도를 보이고 있다. 클라이언트는 GUI(Graphic User Interface), Display Manager, Lock Manager, Cache Manager, Message Parser, 그리고 Network Manager의 크게 6개의 Manager로 구성되어 있으며 서버는 GUI, Work Manager, Propagate Manager, Disk I/O Manager, Lock Manager, Message Parser, 그리고 Network Manager로 구성되어 있다. 주요 Manager의 기능은 다음과 같다.

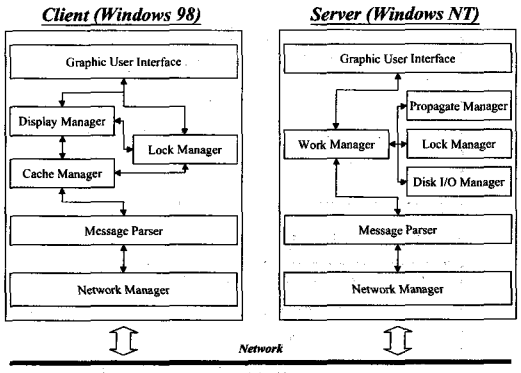


그림 8. 시스템 구성도

6.2 구현 화면

이 절에서는 수치지도 합병 처리기의 실제 구현 화면을 보여 준다.

그림 9는 클라이언트 A가 인접한 클라이언트 B의 작업공간에 있는 엔티티와의 합병 작업을 위해 작업하고자 하는 엔티티가 있는 영역을 확대한 화면을 보이고 있다. 그림 10의 (가)는 클라이언트 A가 클라이언트 B에게 합병 작업을 요청 메시지를 보냈을 때 클라이언트 B의 화면에 출력되는 그림이다. 그림 (나)는 클라이언트 B가 클라이언트 A의 합병 작업 요청에 대해 허락한다는 메시지를 보냈을 때 클라이언트 A의 화면에 출력되는 그림이다. 클라이언트 A는 (나)의 메시지를 받은 이후에 합병 작업을 수행하게 되고, 합병 결과를 서버에 전송하게 되고, 서버는 클라이언트 B 및 관련 클라이언트들에게 변경 메시지를 보냄으로써 하나의 합병 작업이 종료되게 된다. 그림 11에서는 합병된 결과 화면을 보여주고 있다.

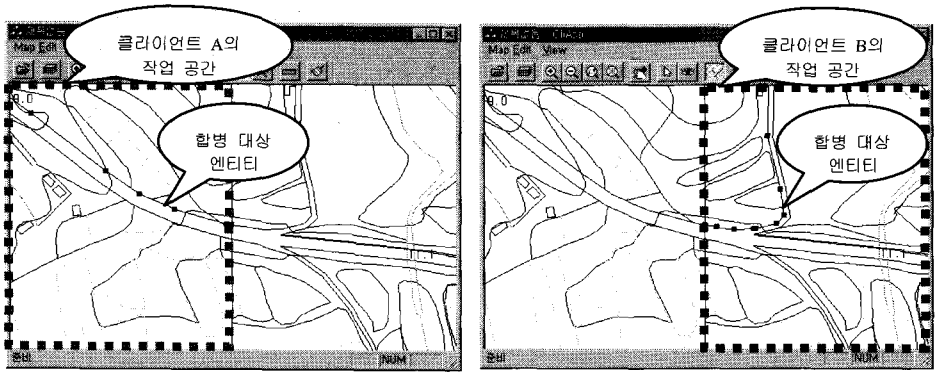
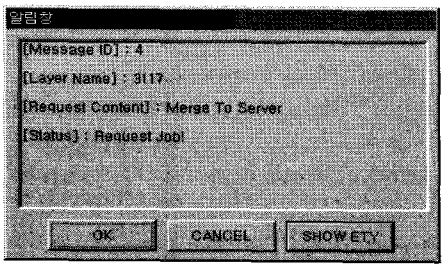


그림 9. 작업 영역 확대 화면

(가) 합병 작업 요청 메시지 박스



(나) 합병 작업 허락 메시지 박스

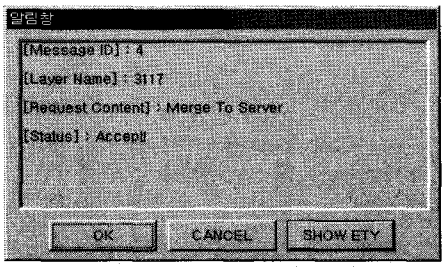


그림 10. 합병 작업 메시지 박스

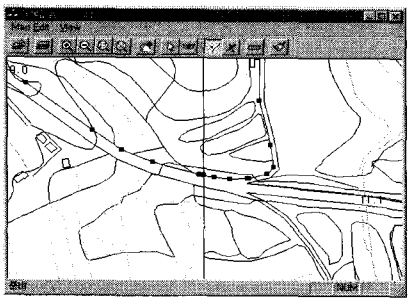


그림 11. 합병 결과 화면

7. 결론 및 향후 연구방향

최근 들어 GIS 구축사례가 점차 증가하는 환경에서, 공간 데이타베이스 구축을 위한 보다 효과적인 방법론에 대한 요구가 더욱 커지고 있다. 공간 데이타베이스 구축을 위해서 기존에 제작되어 있는 수치지도를 이용하는 사례가 점차 증가하는 것도 이러한 필요성에 따른 것이라 할 수 있다. 국립지리원에서 제작한 수치지도의 예와 같이 일반적으로 수치지도는 타일-기반으로 제작되어진다. 대상 지역을 많은 수의 작은 타일로 구분하여 각각에 대해 수치지도를 제작하게 된다. 따라서 이러한 수치지도를 이용할 경우에는 우선 타일로 구분된 수치지도들을 하나로 묶어서 Seamless Map을 만들어야 한다. GIS를 구축하고자 하는 대상 영역이 작은 경우에는 그에 비례해서 수치지도의 수량도 적어지게 되므로 이러한 경우에는 Seamless Map을 제작하는데 큰 비용이 들지 않는다. 하지만 GIS 구축 대상 영역이 넓은 경우에는 Seamless Map을 제작하는데 상당한 시간과 비용이 투입되어야 한다. 이와 같은 경우에는 Seamless Map을 제작하기 위한 보다 효과적인 방법에 대한 연구가 필요하다.

이 논문에서는 타일-기반의 수치지도를 이용하여 Seamless Map을 제작하고자 할 때 작업의 생산성을 높이기 위해 다수의 작업자가 동시에 합병 작업을 수행할 수 있도록 하기 위해 동시 합병 작업 프로토콜 및 그에 따른 세부 알고리즘을 제시함으로써 동시 합병 작업 시에 발생할 수 있는 여러 문제점들을 해결할 수 있게 되었다. 또한 이를 실제 구현해 보임으로써 수치지도에 대한 동시 합병 기법이 유용함을 보였다.

그러나, 보다 완성된 형태의 수치지도 동시 합병 처리기를 위해서는 향후 추가적인 연구가 이루어져야 한다. 향후 연구 과제로는 다음과 같은 세 가지로 볼 수 있다.

첫째, 구현과정에서 고려되지 않았던 메시지 패킷의 최적화 및 메시지 트래픽의 최적화 전략에 대한 연구가 필요하다. 이 논문에서 제시하고 있는 프로토콜은 클라이언트와 서버간의 메시지 교환의 오버헤드가 크므로 전송되는 패킷의 내용에 대한 최적화 작업이 필요하고 또한 메시지 교환에 있어서도 최적화가 이루어짐으로써 불필요한 메시지 교환이 발생하지 않도록 하는 방법에 대한 연구가 필요하다.

둘째, 이 논문에서는 클라이언트의 로컬 디스크에 존재하는 수치지도에 대한 데이타 수정이 아니라, 메모리 캐시 상의 데이타에 대한 교체만을 고려하였다. 따라서 디스크의 수치지도에 대한 캐시 교체 전략에 대한 연구가 추가적으로 이루어져야 한다. 마찬가지로 서버의 경우에는 로그파일에 변경된 데이타의 기록만을 기록하는 것으로 구현은 완료되어 있다. 로그파일과 디스크 상의 수치지도의 Merge에 대한 방법도 추가적으로 연구되어야 한다.

마지막으로, 이 논문에서는 수치지도의 엔티티들이 planar-graph라고 가정함으로써 합병 작업시에 발생할 수 있는 2차 수정 작업에 대해서는 언급하지 않았다. 그러나, 실제 수치지도에서는 planar graph 뿐만이 아니라 non-planar graph인 엔티티들도 상당수 발견되기 때문에 합병 처리기를 실제 현장에서 응용하기에는 한계를 가질 수 밖에 없다. 그러므로, 보다 완성된 형태의 합병 처리기를 위해서는 non-planar graph인 수치지도에 대한 합병 작업 시 발생할 수 있는 2차 수정 작업에 대한 연구가 추가적으로 필요하다.

참고문헌

[1] Christopher B. Jones, "Geographic Information Systems and Computer Cartography", 1997.

[2] 신영상, 최진오, 홍봉희, "클라이언트-서버 환경에서 캐쉬 공간 데이타의 변경 전파", 한국정보과학회 '99 봄학술발표논문집(B), 제 26권 1호, pp 86-88.

[3] AmSuk Oh, JinOh Choi, BongHee Hong, "An Incremental Update Propagation Scheme for a Cooperative Transaction Model", Preceedings of the International Conference on DEXA, 1996.

[4] Korth H. F., Speegle, G. D., "Long-Duration Transactions in Software Design Projects", Proceedings of the International Conference on Data Engineering, pp.586-574, 1990.

[5] Korth H. F., "On Long-Duration CAD Transaction", ACM Transactions on Information Systems, 1987.

[6] Korth H. F., "A Model of CAD Transactions", Proceedings of VLDB Conference, 1985

[7] Edward T. Blair, "Version Management in the Work Order Processing Environment", AM/FM International, 1994

[8] Xin Chang Zhang, "Geometric Feature-based Edge-Matching", Proceedings of the 3rd Internatioanl Conf. on GeoComputation, 1998

[9] JinOh Choi, YoungSang Shin, BongHee Hong, "Update Propagation of Replicated Data in Distributed Spatial Databases", International Conference on DEXA 99, 1999

[10] 국립지리원, http://www.nig,go.kr

[11] Marian H. Nodine, Stanley B.Zdonik, Cooperative Transaction Hierarchies: A Transaction Model to Support Design Applications, Proceedings of the 16th VLDB Conference, 1990

[12] Henry F. Korth, Gregory D. Speegle, Long-Duration Transactions in Software Design Projects, Proceedings of the 6th International Conference on Data Engineering, 1990

[13] Hamideh Afsarmanesh, Dennis McLeod, An Extensible Object-Oriented Approach to Databases for VLSI/CAD, Proceedings of VLDB Conference, 1985

[14] Michael J. Franklin, Michael J. Carey, Miron Livny, Local Disk Caching for Client-Server Database Systems, Proceedings of the 19th VLDB Conference, 1993

[15] Michael J. Franklin, Michel J. Carey, Client-Server Caching Revisited, Proceedings of the International

Workshop on Distributed Object Management, 1992
[16] Michael J. Franklin, Donald Kossmann, Cache Investment Strategies , Univ. of MD Technical Report CS-TR-3803 and UMIACS-TR-97-50, 1997
[17] Thomas M. Lillesand, Ralph W. Kiefer, Remote Sensing and Image Interpretatio n", John Wiley & Sons, Inc, Third Edition, 1994
[18] Butler Lampson, David Lomet, "A New Presumed Commit Optimization for Two Phase Commit", Proceedings of the 19th VLDB Confrence, 1993
[19] 柳福模, "航空寫眞測量 新技術 調査研究", 國土開發研究院, 1996
[20] 김계현 "GIS 개론", 大英社, 1998
[21] Thomas M. Lillesand, Ralph W. Kiefer, "Remote Sensing and Image interpretation", John Wiley & Sons, Inc., 1995

저자 약력

이 상 현
1992년 부산대학교 기계공학과 졸업(공학사)
1998년- 현재 부산대학교 대학원 지형정보협동과정, 석사과정
관심분야: GIS, 공간 데이타베이스, 개방형지리정보시스템

김 동 현
1995년 부산대학교 컴퓨터공학과(학사)
1997년 부산대학교 컴퓨터공학과(석사)
1998년-현재 부산대학교 컴퓨터공학과, 박사과정
관심분야 : 개방형 GIS, 분산공간데이타베이스, 공학 데이타베이스

홍 봉 희
1982년 서울대학교 전자계산기공학과 졸업(공학사)
1984년 서울대학교 대학원 전자계산기공학과 졸업 (공학석사)
1988년 서울대학교 대학원 전자계산기공학과 졸업 (공학박사)
1987-현재 부산대학교 공과대학 컴퓨터공학과 교수
관심분야: 공간 데이타베이스, GIS표준화, 병렬 GIS, 개방형지리정보시스템