다중 미들웨어 컴포넌트를 위한 공간 데이타의 공유 캐싱 기법†

(Shared Caching of Spatial Data for Multiple Middleware Components)

박경미*, 안경환**, 홍봉희***
(Park KyungMi*, An KyoungHwan**, Hong BongHee***)

초 록

현재 지리정보 기술의 세계적인 추세는, ISO 나 OGC 등의 국제 표준기구에서 제시하고 있는 상호 운용성을 지원하기 위한 3 계층 구조의 인터넷 GIS 로 나아가고 있다. 특히 OGC 에서는 인터넷 환경에서 표준 인터페이스를 통해 다양한 데이타 소스의 지리정보를 접근하여 기존의 웹 브라우저로 표현하는 구조의 웹 매핑 테스트베드를 제시하고 있다.

3 계층 구조의 인터넷 GIS 에서의 문제점은 각 계층간의 통신 횟수와 데이타 전송량이 많고, 미들웨어 컴포넌트에서의 데이타 변환으로 인해 속도가 저하되는 것이다. 데이타 전송량과 데이타 변환량으로 인한 문제점은 미들웨어 계층에서 공간 데이타를 캐싱함으로써 해결될 수 있다. 본 논문의 핵심 아이디어는 미들웨어 컴포넌트들 사이에 공유 캐쉬를 두고, 여러 클라이언트들이 공통적으로 접근하는 데이타를 중심으로 캐쉬를 관리하는 것이다.

본 논문에서 제시하는 기법은 질의 영역에 대해 확장된 영역을 캐싱함으로써, 하나의 클라이언트 입장에서 근접한 영역의 질의에 대한 사용자 응답 시간을 줄일 수 있다. 또한 캐쉬 교체를 위해 접근 빈도수 중심의 교체 함수를 취함으로써 여러 클라이언트들에 대해 데이타의 재사용성을 높일 수 있다.

키워드

ISO, OGC, 인터넷 GIS, 웹 매핑 테스트베드, 미들웨어, 컴포넌트, 캐싱, 공유 캐쉬

---

† 본 연구는 정통부의 대학기초연구지원사업의 연구비 지원으로 이루어졌음.
* 부산대학교 GIS 학과 석사과정
** 부산대학교 컴퓨터공학과 박사과정
*** 부산대학교 컴퓨터공학과 교수

1. 서론

현재 GIS 분야에서는 클라이언트가 표준 인터페이스를 제공하는 미들웨어를 통해 기존의 다양한 데이타 소스의 지리 정보에 접근하는 3 계층 구조의 GIS 표준에 대한 연구가 활발히 진행되고 있다[1][2][8]. 인터넷 GIS 환경에서도 3 계층 구조가 채택되고 있으며, 특히 OGC 의 웹 매핑 테스트베드는 OpenGIS 의 표준 인터페이스를 통해서 다양한 데이타 소스에 접근하는 구조의 인터넷 GIS 를 위해 제안되었다[3]. 그러나, 웹 매핑 테스트베드는 표준 인터페이스를 통한 데이타 접근만을 목적으로 하기 때문에 각 계층간의 통신량과 데이타 변환량으로 인한 사용자 응답 시간의 지연 문제는 고려하지 않고 있다.

본 논문에서는 GIS 표준이 적용된 인터넷 GIS 환경에서 전체적인 사용자 응답 시간을 줄이기 위한 미들웨어 측의 캐쉥 기법을 제시하고자 한다. 데이타 소스와 미들웨어 간의 통신량을 줄이고 미들웨어에서의 데이타 변환량을 줄임으로써 여러 클라이언트들에 대해 평균적인 사용자 응답 시간의 개선을 가져올 수 있다.

본 논문에서 제안하는 캐쉥 기법는 다음과 같은 특징을 갖는다. 첫째, 기존의 시맨틱 영역 단위의 방법을 응용하여, 하나의 레이어를 일정하게 분할한 셀 영역의 단위로 질의 결과를 유지한다. 왜냐하면 대상 환경에서의 미들웨어는 데이타 소스에 질의를 통해서만 결과 데이타를 받을 수 있으며, 기존의 페이지나 객체 단위를 요구하여 결과를 얻는 방식은 불가능하기 때문이다. 또한, 레이어를 일정하게 분할하는 것은 캐쉬를 효율적으로 관리하고 캐쉬 단위에 대한 관리 비용을 줄이기 위한 방법이다. 둘째, 미들웨어 컴포넌트간에 공유 캐쉬를 사용하기 위한 구조를 제시한다. 공유 캐쉬를 사용하는 구조는 하나의 클라이언트가 캐쉬를 관리하고 사용하는 구조와는 다른 설계 방법이 필요하다. 셋째, 많은 클라이언트들이 공통적으로 요구하는 데이타를 중심으로 캐쉥한다. 이것은 데이타에 대한 접근 빈도수를 중심으로 한 교체 전략을 적용함으로써 가능하다. 또한 최근 사용 시간을 추가 정보로 사용하여 캐쉬의 효율성을 높인다.

본 논문의 구성은 다음과 같다. 2 장에서는 기존의 캐쉥 기법을 살펴 보고, 3 장에서는 본 논문의 대상 환경, 사용자 응답 시간 지연 문제, 그리고 기존의 캐쉬 교체 전략을 적용할 경우의 문제점을 살펴 본다. 4 장에서는 본 논문에서 제시하는 공유 캐쉬를 사용하는 미들웨어 구조와 질의 결과 생성 시나리오에 대해 설명한다. 5 장에서는 캐쉬 교체 전략에 대해 설명한다. 마지막으로 6 장에서는 결론 및 향후 연구 과제에 대해 기술한다.

2. 관련연구

기존의 캐쉬 기법에 대한 연구는 크게 세 가지로 나뉜다. 첫째, 캐쉬의 적중률을 높이기 위한 캐쉬 교체 전략에 대한 연구가 있다. 둘째, 캐쉬를 효율적으로 사용하기 위한 캐쉬 관리 구조에 관한 연구가 있다. 마지막으로, 페이지나 객체, 시맨틱 영역 등의 다양한 캐쉬 단위에 대한 비교 실험 연구가 있다[4].

캐쉬 교체 전략으로는 데이타의 최근 사용 시간 정보를 기반으로 한 LRU 와 MRU 알고리즘과 데이타의 접근 빈도수를 이용한 LFU 기법이 있으며, 사용자의 접근 유형을 고려하여 객체들간의 공간 관련성을 기반으로 한 알고리즘이 있다. 그리고 LRU 기법을 확장하여 사용빈도수를 추가 정보로 이용하는 LRU-K 알고리즘[6]과 LRU 의 문제점을 보완하기 위해 공간 관련성을 추가 정보로 이용하는 방법[9]이 있다.

캐쉬 관리 구조에 관한 연구로는 서로 다른 두개의 캐쉬를 가지고 데이타의 특성에 따라 관리하는 이중 버퍼 알고리즘[7]이 있다. 그리고, 캐쉬 구조는 하나를 사용하더라도 데이타의 특성에 따라 분리해서 관리하는 방법[6]이 있다.

캐쉬 단위에 따른 연구[4]에서는 각 단위별로 장,단점과 적용될 수 있는 구조에 대해 비교 설명한다. 첫째, 페이지 단위의 캐쉥은 데이타 페이지의 클러스터링 정보를 이용하여 디스크 I/O 를 줄이기 위한 것으로 클러스터링 상태와 인덱스에 따라 캐쉬의 효율성이 달라진다. 둘째, 객체 단위의 캐쉥은 최대의 융통성을 가지지만 각 객체 단위별로 관리하는 메모리 비용이 크며 데이타를 요청하는 통신횟수가 많기 때문에 성능이 떨어지게 된다. 셋째, 시맨틱 캐쉥은 클라이언트의 질의에 대한 결과를 단위로 하기 때문에 객체 단위에 비해 관리 비용이 적으며 페이지 단위에 비해 캐쉬 단위 내의 데이타는 의미적으로 연결성을 가질 수 있다.

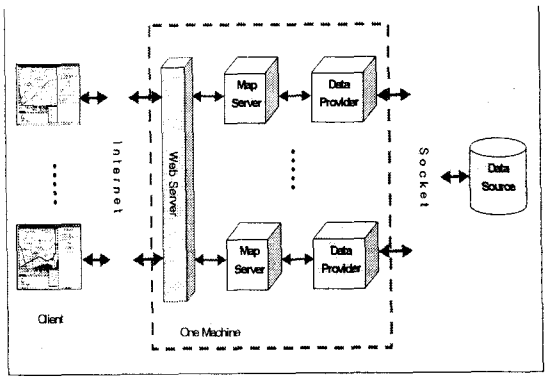
이러한 기법들은 2 계층(클라이언트-서버) 구조에서 클라이언트의 캐쉬 적중률을 높이거나 서버에서 디스크 I/O 를 줄일 수 있으나, 3 계층(클라이언트-미들웨어-데이타 소스) 구조에서 여러 클라이언트를 고려한 미들웨어 단계의 캐쉬에 적용하기 어렵다. 특히, 기존의 캐쉬 교체 전략은 하나의 클라이언트만을 고려하였기 때문에 컴포넌트 기반의 미들웨어에서 적용하는데 문제점이 있다.

3. 대상 환경 및 문제 정의

이 장에서는 본 논문의 연구 대상 환경에 대해 설명하고 이러한 환경에서 사용자 응답 시간 지연의 개선 방법에 대해 설명한다. 그리고 기존의 캐쉬 교체 전략을 적용할 경우의 문제점을 제시한다.

3.1 인터넷 지리정보 서비스 환경

본 논문에서는 [그림 1]과 같이 인터넷 기반의 클라이언트-미들웨어-데이타 소스 구조를 연구 대상으로 한다.



[그림 1] 인터넷 지리정보 서비스 환경

- 클라이언트 : 데이타 소스의 종류에 관계없이 미들웨어에서 제공하는 표준 인터페이스를 통해 공간 데이타를 접근한다. 미들웨어와는 HTTP 와 같은 인터넷

통신 프로토콜을 이용하여 원하는 영역의 맵을 요청하고 XML 데이타를 전송받는다. XML 데이타를 해석하여 기존의 웹 브라우저로 지도를 출력한다.

- 미들웨어 : 웹 서버, 맵 서버, 데이타 제공자로 구성되며 맵 서버나 데이타 제공자는 표준 인터페이스를 제공하기 위해 데이타를 변환한다. 웹 서버는 요청된 질의에 대하여 맵서버에서 만들어진 XML 데이타를 클라이언트에 전송한다. 맵 서버는 해당 데이타 소스를 위한 데이타 제공자를 이용하여 데이타 소스로부터 영역 질의의 결과를 전송받아 XML 데이타로 변환한다.
- 데이타 소스 : 기존의 GIS 서버로서 고유의 통신 프로토콜을 가지기 때문에 서로 다른 인터페이스로 지리 정보를 제공한다. GIS 서버는 공간 데이타에 대한 영역 질의의 결과를 효율적으로 생성하기 위해 공간 데이타의 인덱스를 구성할 수 있으며, 요청된 질의 영역에 대해 벡터 형태의 결과를 미들웨어로 전송한다.

3.2 사용자 응답 시간 개선 방법

이 절에서는 대상환경에서 인터넷 GIS 클라이언트의 질의에 대한 사용자 응답 시간의 개선하기 위한 방법에 대해 살펴 본다.

우선, 질의 Q 에 대한 사용자 응답 시간은 다음과 같이 계산될 수 있다.

$$T(Q) = T_C + T_{CM} + T_M + T_{MS} + T_S$$

- T_C : 클라이언트에서 XML 데이타를 지도로 보여주기 위해 데이타를 변환하고 화면에 출력하는 시간이다.
- T_{CM} : 클라이언트와 미들웨어간의 통신 비용으로서, 질의 영역의 맵을 요청하고 결과 데이타를 받아오는 시간이다.
- T_M : 표준 인터페이스를 제공하기 위해 맵서버나 데이타 제공자에서 데이타를 변환하는 시간이다.
- T_{MS} : 미들웨어와 데이타 소스간의 통신 비용으로서, 질의를 보내고 결과 데이타를 패킷으로 받는 시간이다. 특히, 요청 질의가 여러 개의 레이어에 대한 것일 경우 레이어별로 분할해서 요청해야 하므로 레이어 개수에 따라 통신 횟수와 통신량이 많아질 수 있다.
- T_S : 데이타 소스에서 현재 질의 영역에 해당하는 공간 데이타 객체를 검색하여 결과를 생성하는 시간이다.

현재 대상환경에서 위의 사용자 응답 시간을 줄이기 위한 방안으로 클라이언트, 미들웨어, 데이타 소스의 각 단계에서 캐싱을 고려할 수 있다. 첫째, 클라이언트의 캐싱의 경우 하나의 클라이언트 입장에서 볼 때 사용자 응답 시간 T(Q)를 줄이기 위한 최선의 방법일 수 있다. 특히 사용자 접근 유형을 고려한 공간 관련성이나 LRU 와 같은 기존의 연구 방법을 적용할 수 있다. 둘째, 미들웨어 캐싱은 클라이언트의 캐싱과는 독립적으로 여러 사용자들에 대해 T_M, T_{MS}, T_S 를 줄임으로써 평균적인 사용자 응답 시간을 줄일 수 있다. 셋째, 데이타 소스 캐싱의 경우 T_S 를 줄이기 위해 효율적인 공간 데이타 인덱싱과 병합하여 사용될 수 있다. 그러나,

데이타 소스 캐쉉은 대상 환경의 문제점인 계층간의 통신량과 데이타 변환량으로 인한 지연 문제는 해결하지 못한다.

3.3 문제 정의

본 논문에서는 사용자 응답 시간을 줄이기 위해 미들웨어 캐쉉을 제안한다. 이 절에서는 기존의 교체 전략을 본 논문의 대상 환경에 적용할 경우의 문제점을 설명한다.

3.3.1 공간 관련성 중심의 교체 전략의 문제점

공간 관련성 중심의 교체 전략은 하나의 클라이언트의 질의에 대해 접근 유형을 고려하여 공간 근접성이 있는 데이타를 중심으로 캐쉉한다. 교체 함수는 현재 질의 영역에 대한 캐쉬된 영역과의 거리를 이용한다. 교체 값은 거리에 비례하며 값이 큰 영역이 교체 대상으로 결정된다. 현재 질의 영역의 중심점이 (x_u, y_u)이고 캐쉬된 영역 S의 중심점이 (x, y)인 경우 캐쉬된 영역 S의 교체 값 RP는 다음과 같이 계산될 수 있다.

$$RP = |x - x_u| + |y - y_u|$$

계산된 교체 값이 가장 큰 영역부터 교체 대상으로 결정된다. 대상환경에서 이 기법을 적용할 경우는 하나의 클라이언트에 대한 미들웨어 컴포넌트에서 적용될 수 있으나, 이런 경우는 클라이언트에서 캐쉉하는 것과 다를 바 없다. 또한, 컴포넌트 기반의 미들웨어는 생성, 소멸 주기가 짧기 때문에 하나의 컴포넌트 내의 캐쉉은 의미가 없다. 여러 클라이언트를 고려할 경우, 공간 관련성 중심의 교체 전략을 적용한다면 공간 근접성의 기준을 어떻게 결정할 것인가가 문제이다. 즉, 기준이 되는 현재 질의를 결정하기 위해 하나의 클라이언트를 선택할 방법이 없다.

3.3.2 LRU 기법의 문제점

LRU 기법은 현재 캐쉬된 영역 중에서 최근 사용 시간이 가장 오래된 영역을 교체 대상으로 정한다. 캐쉬된 영역 S의 교체 값 RP는 다음과 같이 계산될 수 있다.

RP = 현재 시간 − S의 최근 사용 시간

이 기법의 문제점은 여러 클라이언트에서 빈번하게 사용되는 영역이 접근 빈도수가 낮은 영역에 비해 최근 사용 시간가 오래된 경우 교체 대상으로 결정된다는 것이다. 따라서, 이 기법은 대상 환경에서 여러 클라이언트를 고려하는 교체 전략으로 적용되기에는 부적합하다.

3.3.3 LRU-K 기법의 문제점

LRU-K 기법은 LRU 기법을 확장한 것으로 사용 빈도수를 고려한 캐쉬 교체 전략이다. 이 기법은 캐쉬된 영역들에 대한 접근 시간 정보 리스트$(r_1, r_2, \ldots, r_t)$를 유지하여 캐쉬된 영역이 현재 시간에서 일정 빈도수 K번 이전에 사용된 시간을 이용하여 교체 값을 결정한다. 현재 시점 t에서 캐쉬된 영역 S의 교체 값 RP_t는 다음과 같다.

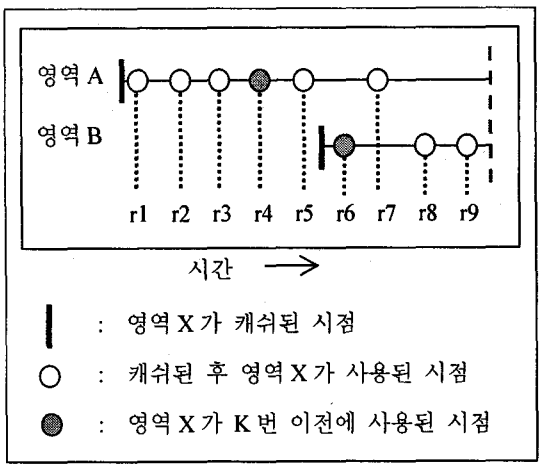
S가 일정 빈도수 K번 이상 접근된 경우 K번 이전의 접근 시간 정보가 r_m이면,

$$RP_t(S,K) = t - m$$

S가 K번 미만으로 접근된 경우,

$$RP_t(S,K) = \infty.$$

이 기법의 문제점은 [그림 2]에서 볼 수 있듯이, K가 3이고 캐쉬된 영역 중 K번 미만으로 접근된 영역은 없다고 가정하면 현재 시점 t = 9 에서 캐쉬된 영역 A 의 교체 값은 5 이고 영역 B 의 교체 값은 3 이 되어 교체 대상은 영역 A 로 결정된다는 것이다. 즉, 아무리 접근 빈도수가 높은 영역이더라도 K 번 이전의 사용 시간에 따라 교체 대상으로 결정될 수 있다는 것이다.



[그림 2] 시간에 따른 캐쉬 셀의 접근 1

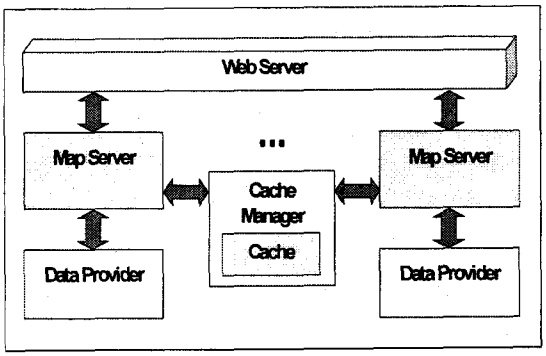
4. 전체 구조 및 시나리오

본 논문에서 제시하는 캐쉥 기법을 적용하기 위한 미들웨어 캐쉬 구조와 그것을 이용한 질의 생성 방식을 살펴보기 전에 미들웨어 캐쉥에서 고려되어야 할 사항을 언급한다.

4.1 고려 사항

본 논문의 대상 환경에서 클라이언트는 다양한 데이타 소스의 공간 데이타를 접근할 수 있다. 따라서, 미들웨어에서 캐쉥할 대상이 되는 데이타를 결정해야 한다. 즉, 요청된 모든 데이타 소스에 대해 캐쉥할 것인지, 하나의 데이타 소스에 대해서만 캐쉥할 것인지를 결정해야 한다. 본 논문에서는 하나의 데이타 소스에 대해 캐쉥하는 것으로 가정한다. 다양한 데이타 소스에 대한 캐쉥을 위해서는 여기서 제시하는 방법을 확장해서 사용하는 방안이 검토되어야 한다.

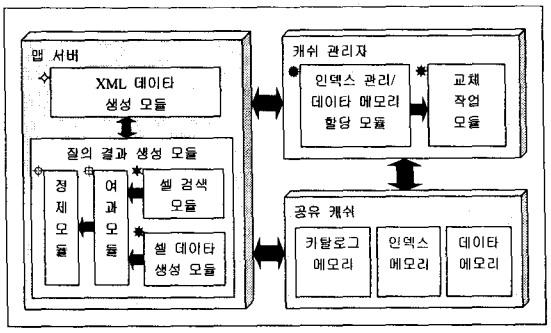
그리고, 미들웨어에서는 웹서버, 맵 서버, 데이타 제공자로 구성되기 때문에 어떤 단계의 컴포넌트들이 캐쉬를 공유해서 사용할 것인가를 결정해야 한다. 웹 서버는 단지 맵을 요청하고 생성된 XML 데이타를 전송하는 단계이므로 캐쉥의 대상으로 고려하지 않는다. 공유 캐쉬를 맵 서버 컴포넌트에 둘 경우가 데이타 제공자에 둘 경우보다 이점이 있다. 왜냐하면 캐쉬된 데이타만으로 질의 결과를 생성할 수 있을 경우 데이타 제공자를 생성하는데 드는 비용을 줄일 수 있을 뿐만 아니라, 데이타 제공자의 표준 인터페이스를 통해 가져온 데이타 형태를 기준으로 한 알고리즘은 다양한 데이타 소스에 대해 확장할 경우 더 쉽게 적용될 수 있기 때문이다.



[그림 3] 공유 캐쉬를 사용하는 구조

4.2 캐싱을 고려한 미들웨어 구조

본 논문에서 제시하는 공유 캐쉬를 사용하는 미들웨어의 전체 구조는 [그림 3]과 같다. 앞에서 언급했듯이 맵 서버 컴포넌트 간에 캐쉬를 공유하는 미들웨어 구조이다. [그림 4]는 공유 캐쉬, 캐쉬 관리자, 맵 서버의 세부적인 설계 구조와 그 관계를 보여준다.



[그림 4] 맵서버, 캐쉬, 캐쉬 관리자 구조

4.2.1 공유 캐쉬의 구조

공유 캐쉬는 카탈로그 메모리, 인덱스 메모리, 데이타 메모리로 구성된다.

- 카탈로그 메모리

데이타 소스의 레이어에 대한 정보를 유지하기 위한 메모리이다. 각 레이어별로 레이어 이름, 전체 영역의 시작과 끝 좌표(StartX, StartY, EndX, EndY), 레이어를 구성하는 칼럼에 대한 정보 등의 레이어의 메타 정보가 저장된다. 또한, 레이어의 전체 영역의 X 축/Y 축 셀 분할 수(XRes, YRes), 분할된 셀의 X 축/Y 축 길이(CellSizeX, CellSizeY) 등의 셀 분할 정보가 저장된다.

- 인덱스 메모리

각 레이어별로 분할된 셀 데이타를 관리하기 위한 인덱스로 구성되는 메모리이다. 각 레이어에 대해 XRes × YRes 의 2 차원 배열 형태로 셀에 대한 인덱스가 저장된다. 캐쉬된 셀에 대해서는 데이타 메모리에서 셀 데이타의 위치 정보와 셀 데이타의 길이가 저장된다. 그리고, 각 셀 데이타의 캐쉬된 시간, 현재 사용자수, 사용 빈도수, 최근 사용 시간 등의 교체 작업을 위한 정보가 유지된다.

- 데이타 메모리

각 캐쉬된 셀의 실제 데이타가 저장되는 메모리로써 데이타 소스의 총 데이타량의 10%를 저장할 수 있는 크기라고 가정한다. 맵 서버의 셀 데이타 생성 모듈에서 생성된 데이타의 길이만큼의 공간이 할당된다.

4.2.2 맵 서버의 구조

맵 서버는 질의 결과 생성 모듈과 XML 데이타 생성 모듈로 구성된다. 질의 결과 생성 모듈은 셀 검색 모듈, 셀 데이타 생성 모듈, 여과 모듈, 정제 모듈로 구성된다.

- 셀 검색 모듈

질의 영역에 해당하는 셀을 조사하여 캐쉬된 셀의 리스트와 캐쉬되어야 할 셀의 리스트를 반환한다.

■ 입력 : 레이어 이름, 질의 영역 좌표
(Xlow, Ylow, Xhigh, Yhigh)

■ 출력 : 캐쉬된 셀 리스트
CL = {(C0i,C0j),...,(Cmi,Cmj)},
캐쉬될 셀 리스트
NCL = {(NC0i, NC0j),...,(NCni,NCnj)}

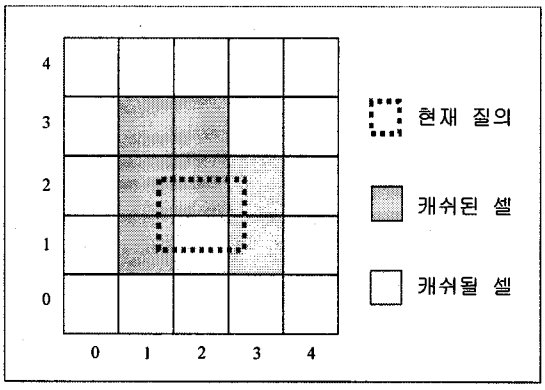
■ 절차 :

1. 카탈로그의 레이어 분할 정보를 이용하여 해당 셀의 시작 인덱스와 끝 인덱스를 계산한다.

CellMinX = (Xlow – StartX) / CellSizeX
CellMaxX = (Xhigh – StartX) / CellSizeX
CellMinY = (Ylow – StartY) / CellSizeY
CellMaxY = (Yhigh – StartY) / CellSizeY

2. For i = CellMinX to CellMaxX
   2.1 For j = CellMinY to CellMaxY
      2.1.1 If 셀 C(i,j)가 캐쉬된 셀이면
         Then CL 에 (i,j)를 추가하고,
            C(i,j)의 사용자수를 1 증가시킨다.
         Else NCL 에 (i,j)를 추가한다.

[그림 5]와 같이 질의가 주어질 경우 이 모듈의 결과는 CL = {(1,1), (1,2), (2,2)}, NCL = {(2,1), (3,1), (3,2)}이다.



[그림 5] 질의 예제

- 셀 데이타 생성 모듈

캐쉬되어야 할 셀에 대해 데이타 제공자를 통해 셀 데이타를 생성하여 캐쉬에 저장한다.

■ 입력 : 레이어 이름, 캐쉬될 셀 리스트
NCL = { (NC0i, NC0j),...,(NCni,NCnj)}

■ 절차 :

1. For k = 0 to n
   1.1 셀 데이타의 영역 좌표를 계산한다.
      Xmin = StartX + NCki * CellSizeX
      Ymin = StartY + NCkj * CellSizeY
      Xmax = StartX + (NCki + 1) * CellSizeX
      Ymax = StartY + (NCkj + 1) * CellSizeY
   1.2 데이타 제공자를 이용하여 해당 레이어의 영역 (Xmin, Ymin, Xmax, Ymax)의 데이타를 가져온다.
   1.3 1.2 의 결과를 캐쉬에 저장하기 위해 캐쉬 관리자에서 셀 데이타의 크기만큼 데이타 메모리를 할당 받는다.
   1.4 할당 받은 메모리에 1.2 의 데이타를 저장한다.
   1.5 교체 대상에서 제외시키기 위해 셀의 사용자수를 1 로 설정한다.

- 여과 모듈

질의 영역에 해당하는 캐쉬된 셀 리스트를 이용하여 각 셀 데이타에서 질의 영역에 포함되는 객체를 추출한다.

■ 입력 : 레이어 이름, 질의 영역 좌표, 셀 리스트 {(i0, j0),...,(ik, jk)}
■ 출력 : 셀 데이타에서 추출된 질의 영역 내의 후보 객체 집합 S1, S2
■ 절차 :

1. For h = 0 to k
   1.1 For each object in C(ih,jh)
      1.1.1 If object 의 MBR 이 질의 영역에 포함되면
         Then S1 에 객체를 추가한다.
         Else
            If object 의 MBR 이 질의 영역의 boundary 와 교차되면
            Then S2 에 추가한다.
   1.2 셀 C(ih,jh)의 사용빈도수를 1 증가시키고 사용자수는 1 감소시킨다.

- 정제 모듈

여과 모듈에서 만들어진 중간 결과로부터 질의 영역 내에 완전히 포함되는 객체를 추출하며 중복된 객체를 제거한다.

■ 입력 : 중간 결과 객체 집합의 S1, S2
■ 출력 : 중복 객체가 제거된 질의 영역에 해당하는 모든 객체 집합 S
■ 절차 :

1. 집합 S1 의 모든 객체를 S 에 추가한다.
2. 집합 S2 의 모든 객체에 대하여 2.1 을 반복한다.
   2.1 객체의 모든 점이 질의 영역의 내부에 있으면 객체를 S 에 추가한다.
3. 집합 S 의 모든 객체에 대해 객체 ID 로 Sorting 한다.
4. 3 의 결과에서 ID 가 같은 객체들의 중복을 제거한다.

- XML 데이타 생성 모듈

질의 결과 생성 모듈의 결과에서 클라이언트가 요청한 칼럼을 추출하여 XML 데이타로 만든다.

4.2.3 캐쉬 관리자의 구조

캐쉬 관리자는 인덱스 관리/데이타 메모리 할당 모듈과 교체 작업 모듈로 구성된다.

- 인덱스 관리/데이타 메모리 할당 모듈

맵 서버의 셀 데이타 생성 모듈에서 요구하는 데이타 메모리의 공간을 할당하고 해당 셀의 인덱스에 할당된 메모리 영역의 정보와 셀 데이타 길이를 저장한다.

■ 입력 : 레이어 이름, 셀 인자 (i, j), 셀 데이타 길이
■ 출력 : 할당된 데이타 메모리의 영역 정보
■ 절차 :

1. 데이타 메모리에서 셀 데이타 길이만큼의 빈 공간이 있는지 조사한다.
   1.1 공간이 없으면 교체 작업 모듈을 이용하여 필요한 메모리 공간만큼 비운다.
2. 데이타 메모리에서 셀 데이타 길이만큼 할당한다.
3. C(i,j)의 인덱스 정보에 할당된 데이타 메모리의 영역 정보를 저장한다.

- 교체 작업 모듈

인덱스 관리/데이타 메모리 할당 모듈에서 요구하는 데이타 메모리의 공간을 확보하기 위해 데이타 메모리의 교체 작업을 수행한다.

■ 입력 : 요구되는 데이타 메모리 공간 길이
■ 절차 :
1. 모든 레이어의 캐쉬된 셀들 중에서 현재 사용자수가 0 인 셀을 조사한다.
2. 1 에서 조사된 모든 셀들의 교체값을 계산한다.
3. 교체값이 큰 셀부터 셀 데이타의 길이를 조사하여 요구되는 공간을 만족할 때까지 3.1 에서 3.2 를 반복한다.
3.1 교체 대상 셀의 데이타의 메모리를 해제한다.
3.2 셀의 인덱스 정보를 비운다.

교체 값을 결정하는 함수는 5 장에서 자세히 설명하며, 현재 캐쉬된 셀 중 사용중인 셀의 비율은 일정 비율 이하라고 가정한다. 이 같은 문제에 대한 제어 방법은 향후에 연구할 것이다.

4.3 질의 결과 생성 시나리오

웹 서버를 통해 클라이언트에서 요청된 영역 질의의 결과를 생성하는 과정은 다음과 같다.

① 맵 서버는 셀 검색 모듈을 이용하여 질의 영역에 해당되는 캐쉬된 셀 리스트와 캐쉬되어야 할 셀 리스트를 생성한다.
② 맵 서버는 셀 데이타 생성 모듈을 이용하여 캐쉬되어야 할 셀 리스트의 각 셀에 대하여 데이타 제공자를 통해 셀 데이타를 생성한다.
③ 캐쉬 관리자는 인덱스 관리/데이타 메모리 할당 모듈을 이용하여 맵 서버의 셀 데이타 생성 모듈에서 요청하는 데이타 메모리 공간을 할당한다.
④ 캐쉬 관리자는 교체 작업 모듈을 이용하여 인덱스 관리/데이타 메모리 할당 모듈에서 요청하는 메모리 공간을 확보한다.
⑤ 맵 서버는 여과 모듈을 이용하여 해당 셀에서 중간 결과를 생성한다.
⑥ 맵 서버는 정제 모듈을 이용하여 여과 모듈의 중간 결과에서 중복 객체를 제거한다.
⑦ 맵 서버는 XML 데이타 생성 모듈을 이용하여 질의에 대한 XML 데이타를 생성한다.

5. 캐쉬 교체 알고리즘

이 장에서는 미들웨어 컴포넌트 간에 캐쉬를 공유하는 구조에서 기존의 캐쉬 교체 기법들의 문제점을 해결하고 여러 클라이언트의 질의를 고려한 교체 알고리즘을 제시한다.

5.1 접근 빈도수 중심의 교체 전략

3.3에서 언급한 교체 전략의 문제점은 접근 빈도수 중심의 교체 전략을 이용하여 해결한다. 즉, 여러 클라이언트들에 대한 평균적인 응답 시간을 줄이기 위해 공통적으로 많이 접근되는 셀 데이타를 캐쉬에 남아 있도록 한다.

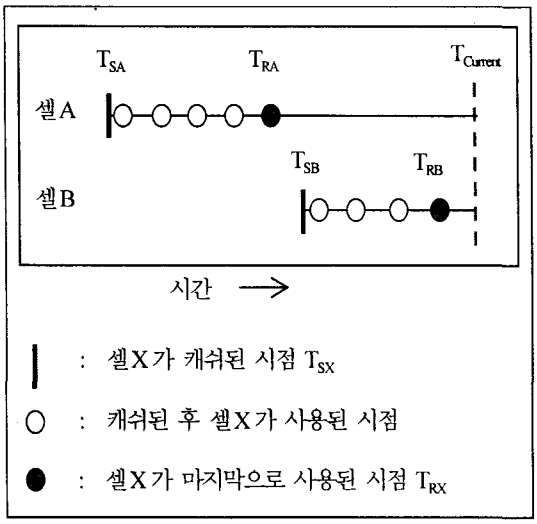
접근 빈도수만을 고려하면, 셀 X 에 대한 교체 값 $C_X.RP$는 다음과 같은 함수를 이용하여 결정된다.

$$C_X.RP = \frac{1}{F_X}$$

(F_X : 셀 X 의 접근 빈도수)

그러나, 이와 같이 접근 빈도수만을 고려할 경우 다음과 같은 문제점을 가진다. 우

선, 셀 X 는 시간 T_{SX}에 캐쉬되어 빈도수 F_X 만큼 접근되었으며 가장 최근에 접근된 시간이 T_{RX}라고 하자.



[그림 6] 시간에 따른 캐쉬 셀의 접근 2

[그림 6]에서는 현재의 시점 $T_{Current}$ 에서 캐쉬된 셀 A 와 셀 B 에 대해서 시간의 흐름에 따라 셀 A 와 셀 B 가 캐싱되어서 접근되는 형태를 보여준다. 셀 A 는 접근 빈도수가 5 이고, 셀 B 는 빈도수가 4 인데, 접근 빈도수만을 고려하면 교체될 셀은 셀 B 로 결정될 것이다. 그러나, 현재의 시점에서 셀 A 에 비해 셀 B 가 계속 사용될 가능성이 높다고 볼 수 있다.

따라서, 최근 사용 시간을 추가 정보로 이용하여 교체 값을 결정할 필요가 있다. 다음 함수는 접근 빈도수를 중심으로 하고 최근 사용 시간을 고려해서 교체 값을 결정한다.

$$C_X.RP = \alpha \frac{1}{F_X} + (1-\alpha)\frac{T_{Current} - T_{RX}}{T_{Current} - T_{SX}}$$

이 함수는 하나의 셀 데이타가 캐쉬에 남아있는 시간, 즉 셀이 캐쉬된 시점과 현재의 시점의 차이에 대한 마지막으로 사용된 후부터 현재까지의 시간의 비율을 사용한다. 셀의 접근 빈도수가 높지만 현재의 시점에서 오랫동안 사용되지 않으면서 캐쉬에 상주할 수 있는 가능성을 배제하고 최근에 사용이 빈번한 셀 데이타를 캐쉬함으로써 캐쉬의 효율성을 높인다. α 값은 접근 빈도수와 최근 사용 시간을 얼마나 고려할 것인가에 따른 가중치로 이용될 수 있으며, 실험을 통해 최적의 값이 결정될 수 있다.

5.2 다중 레이어를 고려한 전략

기존의 캐싱 연구 중에는 캐쉬될 데이타와 그렇지 않은 데이타를 구분해서 선택적으로 캐싱하는 기법이 있다. 이 방법은 프록시 서버 알고리즘에서 웹을 통해 데이타를 가져오는 시간과 데이타 변경 횟수를 기반으로 한다. 즉, 웹을 통해 데이타를 가져오는 시간이 많이 드는 데이타를 캐쉬에 남아 있도록 하거나 데이타의 변경이 잦은 데이타보다 변경이 거의 없는 데이타를 캐쉬하도록 하는 방법이다[5].

본 논문에서는 이러한 기법을 다른 관점에서 적용한다. 사용자 접근 빈도수를 중심으로 하는 전략을 기반으로 사용자의 접근이 많은 레이어를 캐싱 대상으로 한다. 레이어별로 단위 영역을 관리하기 때문에 셀 데이타 관리에 드는 메모리 비용을 무시할 수 없다. 그래서 사용자의 접근이 적고 캐쉬된 셀의 개수가 적은 레이어에 대해서는 캐싱하지 않는 레이어로 등록한다. 캐싱하지 않는 레이어로 등록된 경우, 클라이언트의 요청 영역만을 데이타 소스로부터 가지고 오도록 한다.

6. 결론 및 향후 연구 과제

본 논문에서는 인터넷 GIS 환경에서 미들웨어 컴포넌트들이 캐쉬를 공유하여 클라이언트들에 대한 평균 사용자 응답 시간을 줄이기 위한 캐쉥 기법을 제안하였다. 질의 결과를 중심으로 확장된 영역 단위로 캐쉥함으로써 하나의 클라이언트 입장에서 근접한 지역의 질의에 대한 사용자 응답 시간을 줄일 수 있다. 또한 사용자들이 주로 접근하는 데이타를 위한 사용 빈도수 중심의 교체 전략을 제시하며 캐쉬의 효율적인 사용을 위해 최근 사용 시간을 추가 정보로 이용한다.

향후 본 논문에서 제시한 알고리즘을 적용한 캐쉬 구현 및 성능 평가 실험이 요구된다. 특히, 캐쉬의 크기와 단위 영역의 크기 변화에 따른 실험이 필요하며, 교체 전략에서 최근 사용 시간 정보에 대한 가중치의 변화에 따른 캐쉬의 효율성 검증이 필요하다.

7. 참고 문헌

[1] Open GIS Consortium, The OpenGIS Abstract Specification Model, Version 3, 1999

[2] Open GIS Consortium, The Simple Features Specification for OLE/COM Revision 1.1, 1999

[3] Open GIS Consortium, The Web Mapping Testbed Public Page, 2000 http://www.opoengis.org/

[4] Shaul Dar, Michael J. Franklin, Bjorn T. Jonsson, Divesh Srivastava, and Michael Tan, "Semantic Data Caching and Replacement", In Proceedings of the 22nd Very Large Data Bases, 1996

[5] Junho Shim, Peter Scheuermann, and Radek Vingraleck, "Proxy Cache Algorithms : Design, Implementation, and Performance", IEEE Transactions on Knowledge and Data Engineering, Vol. 11, No. 4, July/August 1999

[6] E.J.ONeil, P.E.ONeil, and G.Weikum, "The LRU-K Page Replacement Algorithm For Database Disk Buffering", In Proceedings of the ACM SIGMOD International Conference on Management of Data, May 1993

[7] A.Kemper and D.Kossmann, "Dual-Buffering Strategies in Object Bases", In Proceedings of the 20 th Conference on Very Large Data Bases, September 1994

[8] 윤우진, "OLE/COM 환경에서 개방형 공간 데이터 서버의 설계 및 구현", 부산대학교 대학원 컴퓨터공학과 석사 학위 논문, 2000

[9] 조대수, 안경환, 홍봉희, "인터넷 지리정보서비스의 성능 개선을 위한 클라이언트 캐쉬 알고리즘", '99 한국 데이타베이스 학술대회 논문집, 제 15 권 1 호, 99. 2.

박 경 미

1995 년 부산대학교 컴퓨터공학과 졸업
(공학사)
1999 년~현재 부산대학교 대학원 GIS 학과,
석사 과정
관심분야 : OLE/COM, 인터넷 GIS, 개방형 GIS

안 경 환

1997 년 부산대학교 컴퓨터공학과 졸업
(공학사)
1999 년 부산대학교 대학원 컴퓨터공학과 졸업
(공학석사)
1999 년~현재 부산대학교 대학원 컴퓨터공학과,
박사 과정
관심분야 : 인터넷 GIS, 분산 객체 기술,
개방형 GIS

홍 봉 희

1982 년 서울대학교 전자계산기공학과 졸업
(공학사)
1984 년 서울대학교 대학원 전자계산기공학과
졸업 (공학석사)
1988 년 서울대학교 대학원 전자계산기공학과
졸업 (공학박사)
현재 부산대학교 공과대학 컴퓨터공학과 정교수
관심분야 : 공간 데이타베이스, GIS 표준화,
개방형 GIS, 병렬 GIS