웹 데이타베이스에서 하이퍼텍스트 모델 확장 및 데이타베이스 게이트웨이의 동적 서버 할당

(Hypertext Model Extension and Dynamic Server Allocation for Database Gateway in Web Database Systems)

신 판 섭 † 김 성 완 †† 임 해 철 †††

(PanSeop Shin) (SungWan Kim) (HaeChull Lim)

요 약 웹 데이타베이스는 웹 환경에서 하이퍼텍스트 모델을 바탕으로 멀티미디어 처리를 위한 부가적인 구조와 관계형 또는 객체지향형 데이타베이스 관리 시스템을 접목하여 구축하는 대용량의 멀티미디어 데이타베이스 응용 시스템이다. 그러나 기존의 하이퍼텍스트 모델링 기법과 DBMS 통로 형태로는 웹 서비스 고급화에 필수적인 다양한 표현능력과 DBMS 연동과정에서의 병목발생으로 인한 동시성 기능이 제한된다. 따라서, 본 논문에서는 하이퍼텍스트 모델링 측면에서 암시적 질의 수행 기능을 지원하고 동적으로 생성되는 항해 모델과 가상 그래프 구조를 제안한다. 또한 항해 유형 분류를 통해 노드와 링크의 생성 규칙을 유도하고 제안된 모델과 웹 데이타베이스 시스템 후위에 위치하는 관계형 모델과의 상호 사상 기법을 연구한다. 그리고 데이타베이스 통로의 효율을 향상시키기 위해 가중치를 기반으로 질의처리 서버를 동적으로 할당하는 스케줄링 기법을 제안하여 시스템 전체의 성능을 개선하고, 제안된 기법이 상대적으로 높은 복잡도를 갖는 동시 질의 요구에 적합함을 보인다.

Abstract A Web database System is a large-scaled multimedia application system that has multimedia processing facilities and cooperates with relational/Object-Oriented DBMS. Conventional hypertext modeling methods and DB gateway have limitations for Web database because of their restricted versatile presentation abilities and inefficient concurrency control caused by bottleneck in cooperation processing. Thus, we suggest a Dynamic Navigation Model & Virtual Graph Structure. The Dynamic Navigation Model supports implicit query processing and dynamic creation of navigation spaces, and introduce node-link creation rule considering navigation styles. We propose a mapping methodology between the suggested hypertext model and the relational data model, and suggest a dynamic allocation scheduling technique for query processing server based on weighted value. We show that the proposed technique enhances the retrieval performance of Web database systems in processing complex queries concurrently.

1. 서 론

최근 들어 웹서비스와 데이타베이스를 통합한 웹 데이타베이스에 관한 연구·개발이 활발하게 이루어지고 있는데, 웹 데이타베이스는 웹의 광대역 서비스 구조와 데이타베이스 시스템의 대용량 자료 관리 기능을 상호 보완한 것으로서 대규모 멀티미디어 서비스 시스템에 적합한 구조를 가지고 있다[1]. <그림 1>은 최근 구축되는 일반적인 웹 데이타베이스 시스템의 구성을 나타낸 것이다 .

웹 데이타베이스 시스템의 기본 모델인 하이퍼텍스트는 사용자가 항해 연산을 이용하여 정보 검색 경로를 생성할 수 있으며, 노드와 링크로 표현되는 단순성 등

· 본 연구는 1998년 한국과학재단 특정기초연구(98-0102-09-01-3)의 지원을 받았음

† 학생회원 : 홍익대학교 컴퓨터공학과
psshin@cs.hongik.ac.kr
†† 비 회 원 : 삼육의명대학 전산정보과 교수
swkim@syu.ac.kr
††† 종신회원 : 홍익대학교 컴퓨터공학과 교수
lim@cs.hongik.ac.kr

논문접수 : 1999년 8월 23일
심사완료 : 2000년 4월 4일

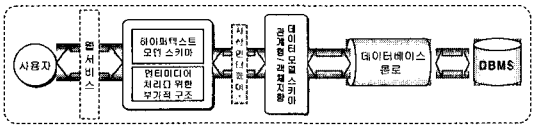


그림 1 일반적 웹 데이타베이스 시스템 구성

[2]으로 인해 전자사전, 도서, 교육지원시스템, 관광, 문서관리, 온라인안내 등 다양한 응용 분야에 적용되고 있다[3]. 그러나 하이퍼텍스트 시스템은 사용자가 항해시, 방향상실 문제, 링크 선택 및 관리의 인지적 부담이라는 고전적 문제점[4] 이외에도 질의 및 검색 지원의 미비[5], 동적 변환 지원기능 취약[2], 모델 자체에 대한 연구 미비[6] 등의 문제점도 지니고 있다. 따라서, 본 연구에서는 질의 기능 및 동적 가상구조 항해 모델을 제안하고, 항해 모델에서 생성되는 동적인 노드와 링크를 정의를 통해 기존의 문제점을 해결한다. 또한 항해 유형 분류를 통해 노드와 링크의 생성 규칙을 유도하고, 동적인 항해 공간을 정의하여 검색자의 다양한 검색을 지원한다.

웹 데이타베이스의 데이타베이스 통로(DB Gateway)는 서비스 성능 측면에서 웹서비스 시스템의 가장 핵심적인 부분으로, 시스템 전체의 검색 성능을 좌우한다[7]. 현재, 데이타베이스 통로에 가장 일반적인 방법으로 CGI(Common Gateway Interface) 응용 서버를 들 수 있는데[8], 이 방법은 기존의 웹서비스와 호환성이나 응용 구축의 용이성 등에서 많은 장점을 보유하고 있으나[3], 연동 과정의 병목발생과 DBMS의 동시성 기능을 전적으로 지원할 수 없다는 문제점을 지니고 있다[9]. 또 다른 방법으로는 API(Application Program Interface)를 이용한 웹서버 확장을 들 수 있는데, 최근 상용 DBMS나 개발 도구들이 많이 사용하는 기법이다. API를 이용한 웹서버 확장은 DBMS 연동 효율을 크게 향상시킬 수 있다는 장점이 있으나[10] 서비스 개발이 어렵고 웹서버가 정의하는 독자적 스크립트를 사용해야 하므로, 기존 웹서비스와의 호환성에서 문제점을 지니고 있다[11]. 따라서 본 논문에서는 호환성과 경제성 면에서 탁월한 CGI 응용 서버 방식을 기반으로 데이타베이스 접근 동시성 향상을 위해 다중 질의처리 서버를 구축하고, 질의처리 서버에서 발생하는 병목현상 문제를 가중치를 기반으로 한 질의처리 서버 스케줄링 기법을 제안하여 해결한다.

본 논문은 다음과 같이 구성된다. 2장에서는 기존의 하이퍼텍스트 모델 및 웹과 데이타베이스의 연동 구조에 관한 관련 연구를 살펴보고, 3장에서는 확장된 하이퍼텍스트 모델을 정의한다. 4장에서는 데이타베이스 통로의 스케줄링 기법에 대하여 기술하고, 5장에서는 성능 평가 실험 결과를 기술하고, 6장에서 결론을 내린다.

2. 관련연구

2.1 하이퍼텍스트 모델 관련 연구

하이퍼텍스트 모델은 고전적 데이타 모델 중에 하나로, 최근에 수행되어진 연구들을 정리하면 다음과 같다.

첫째, 하이퍼텍스트 모델 관련 연구로, [12][13]은 질의 기반 접근을 항해의 보완 요소로 제시하였다. [14]은 하이퍼텍스트 시스템의 질의 기반 접근은 방향상실 해결 및 항해 시작점 발견의 용이성 등을 지닌다고 언급하였다.

둘째, 관계형 데이타베이스 모델에 하이퍼텍스트 모델을 도입하는 연구로, [3]은 두 모델의 통합 개념이 멀티미디어 지원의 용이성 및 항해 연산을 통한 고차원적 데이타베이스 검색에 유리하다고 하였다. [15]은 기존의 데이타베이스 분할 형태의 질의 결과를 링크로 연결하여 연관된 정보를 검색할 수 있도록 하이퍼텍스트의 기능을 도입한 데이타베이스 접근 모델을 설계하였고, [11]은 두 모델을 통합시켜 모델링 기능 및 항해 검색 능력 향상, 노드·링크 자동 생성, 항해 인터페이스 기능 등을 연구하였다. 이러한 연구들은 하이퍼텍스트 모델의 약점을 데이타베이스의 검색 기능 기법을 이용하여 확장하도록 한 연구이며, 편리한 인터페이스, 검색상의 효율성, 데이타베이스의 활용성을 극대화[5] 할 수 있는 장점이 있다.

2.2 데이타베이스 연동기법 연구

웹 데이타베이스는 데이타베이스 통로를 통해 웹서비스와 데이타베이스 사이에 존재하는 이질적인 데이타 관리 기법의 차이를 해소하고 상호 연동 기능을 수행한다. 데이타베이스 통로는 연동 방식에 따라 분류되며[16], 이러한 데이타베이스 통로에 따라 웹서비스의 성능과 구조가 달라지게 된다[17]. 웹과 데이타베이스의 연동 기술은 데이타베이스를 접속하는 프로그램이 웹서버에 위치하는 "서버 확장기법"과 웹 브라우저에 위치하는 "클라이언트 확장기법"으로 크게 분류할 수 있다[8][11]. "클라이언트 확장기법"은 연동 기능이 미약하고 검색자가 별도의 전용 소프트웨어를 사용하여야 하는 단점 때문에, 최근의 연동기법 연구는 대부분이 "서버 확장기법"을 근간으로 수행되어진다. 이러한 "서버 확장기법"은 CGI 응용 서버 방식과 확장 API 방식으로 집약될 수 있으며[8], 이 중에서 응용 확장의 용이성과 CGI 실행 화일 방식의 성능 문제를 해결한 CGI 서버

방식이 주로 사용되고 있다[3].

CGI 응용 서버 방식은 CGI 실행 파일 방식의 데이타베이스 접근 프로세스를 데몬 방식으로 교체한 형태이다[9]. 따라서 불필요한 DBMS 접속 오버헤드를 감소시키고 동시 질의 요구가 발생하여도 시스템 자원 효율을 향상 가능하다. <그림 2>는 CGI 응용 서버 방식을 표현한 것이다.

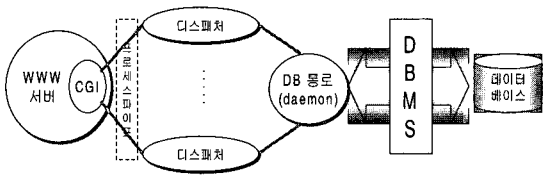


그림 2 CGI 응용 서버 방식

확장 API 방식은 API를 이용하여 웹서버 자체에 데이타베이스 접속 기능을 부가한 형태이다[7]. 따라서 데이타베이스 통로는 웹서버의 API와 DBMS의 API로 구현된 형태를 지니므로 DBMS 전용의 웹서버를 사용하거나 전용의 Script로 연동 응용을 작성한다[10]. 확장 API 방식은 CGI 응용 서버 방식보다 연동 성능과 시스템 자원 활용은 매우 효율적이지만 호환성 및 확장성이 떨어지고 웹서비스 구축 시, DBMS나 확장된 웹서버 전용 Script를 사용해야 하는 문제점이 있다.

본 연구에서는 여러 DBMS 연동기법 연구 중, 기존의 웹 관련 기술을 변경 없이 사용 가능하며 응용 확장이 용이하고 CGI 실행 파일 방식의 성능 문제를 해결한 CGI 응용 서버 방식을 개선하여 DBMS 연동 효율과 동시성 기능을 향상시킨다.

3. 하이퍼텍스트 모델의 확장

하이퍼텍스트 모델은 크게 표현 모델과 구현 모델로 나누어진다. 표현 모델은 정적 데이타 구조를 표현하는 '데이타 서브 모델'과 항해 검색 구조를 표현하는 '프로세스 서브 모델'의 세부 구조를 가지고 있으며, 이러한 표현 모델과 관계형 또는 객체 지향 데이타 모델간의 사상 기법 표현을 구현 모델이라 한다[2].

3.1 데이타 서브 모델 확장

3.1.1 노드의 정의

기존의 정적인 노드는 단순히 노드를 서술하는 슬롯 또는 속성 집합으로 구성된다[6]. 각 슬롯은 이름으로 식별되며 실제 내용을 갖는다. <그림 3>은 기존 하이퍼텍스트 모델의 예를 보인 것이다.

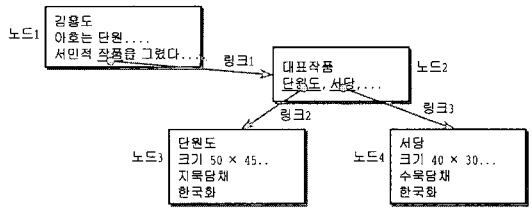


그림 3 기존 하이퍼텍스트 노드·링크 구조의 예

본 연구에서는 노드를 단순히 정보를 저장하는 단위에서 정보들의 집합 또는 그 구조적 명세 정보들의 집합을 저장하는 단위로 확장한다. 따라서 노드의 특성에 따라 스키마 노드, 인스턴스 노드를, 기능에 따라 입력 노드, 출력 노드를 정의한다.

(1) 노드 특성에 따른 분류

[정의1] 스키마 노드(SN : Schema Node)

스키마 노드(SN)는 SN_Name($slot_1$, $slot_2$, ... , $slot_n$)으로 구성된다.

<SN_Name : 노드 이름, {$slot_1$, $slot_2$, ... , $slot_n$} : 슬롯의 집합 리스트>

<표 1-(1)>은 스키마 노드를 도시한 것이며 스키마 노드는 같은 의미와 유형을 공유하는 노드 집합에 대한 구조적 명세 정보로 구성된 노드이다.

[정의2] 인스턴스 노드(IN : Instance Node)

인스턴스 노드(IN)는 스키마 노드 SN_Name($slot_1$, $slot_2$, ... , $slot_n$)에 대한 <$slot\_val_1$, $slot\_val_2$, ... , $slot\_val_n$>로 구성되며, 스키마 노드에 대한 각 인스턴스 노드들의 집합을 인스턴스 노드 집합이라 정의하고 {IN_1, IN_2, ... , IN_n}과 같이 나타낸다.

<$slot\_val_i$: 스키마 노드의 $slot_i$ 에 대한 실제 내용 값>

[제약조건1] 노드 식별자(NID) : 각 인스턴스 노드는 인스턴스 집합 내에서 유일하며, 인스턴스 노드 집합 내에서 각 인스턴스 노드를 유일하게 구별해 주는 슬롯을 노드 식별자라 정의한다.

제약 조건을 고려한 인스턴스 노드는 논리적으로 IN = <INid, other_slot_values>과 같이 기술한다. 인스턴스 노드는 <표 1-(2)>와 같이 표기한다.

(2) 노드의 기능에 따른 분류

- 입력 노드

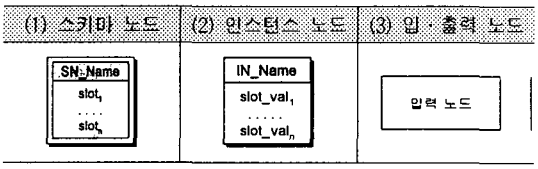
정보 검색자가 애트리뷰트의 값을 할당, 선택하는 노드

- 출력 노드

검색 결과로부터 동적으로 결정되어 구성되는 노드

입·출력 노드의 표시는 <표 1-(3)>과 같다.

표 1 확장 노드의 표시 기호



3.1.2 링크의 정의

본 확장 모델에서는 근원 노드와 목표 노드간의 물리적, 논리적 연결 또는 그 구조적 명세 정보들로 구성된 링크를 정의한다.

(1) 링크의 특성에 따른 분류

[정의3] 스키마 링크(SL : Schema Link)

스키마 링크(SL)는 그 이름 SL_Name(Cardinality, 연관 관계)로 구성된다.

스키마 링크는 스키마 노드간을 연결하는 구조적 명세 정보들로 구성된 동일 의미의 링크이다. 여기에서 SL_Name은 스키마 링크의 유형을 의미하며, 이러한 스키마 링크는 <표 2-(1)>과 같이 표기한다.

[정의4] 인스턴스 링크(IL : Instance Link)

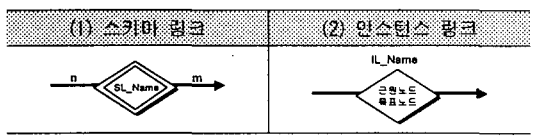
인스턴스 링크(IL)은 스키마 링크 SL에 대한 <근원슬롯, 목표슬롯>로 구성되며, 스키마 링크에 대한 인스턴스 링크 집합을 {IL1, IL2, ... , ILn}과 같이 정의한다.

<근원슬롯, 목표슬롯> : 근원인스턴스 노드의 식별자 INid와 목표인스턴스 노드의 식별자 INid

[제약조건2] 식별자 슬롯 : 인스턴스 링크를 구별하기 위해 (근원 슬롯, 목표 슬롯)쌍의 복합 값으로 식별자 슬롯을 구성하며, 인스턴스 링크의 집합 내에서 각 인스턴스 링크는 유일하다.

인스턴스 링크는 스키마 링크에 대해 실제 값으로 구성된 링크이며, <표 2-(2)>와 같이 표기한다.

표 2 확장 링크의 표시 기호



<그림 4>는 스키마 노드와 스키마 링크로 구성된 스키마 구조의 예를 나타낸 것이다.

그림 4 스키마 노드와 스키마 링크

<그림 4>의 예는 '작가' 스키마 노드, '작품' 스키마 노드와 근원노드·목표노드 링크 슬롯으로 구성된 '작품정보' 스키마 링크로 구성된다. <그림 5>는 <그림 4>의 스키마 구조에 대한 인스턴스 노드·링크를 나타낸 것이다. 작가번호, 작품번호, (근원노드, 목표노드) 쌍은 각각의 식별자 슬롯을 나타내며, 점선으로 표시된 타원은 인스턴스 노드 집합과 인스턴스 링크 집합이다.

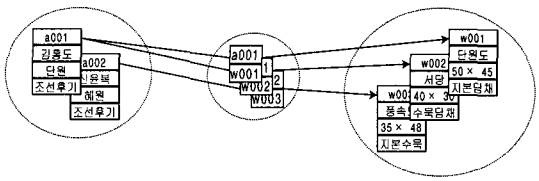


그림 5 인스턴스 노드와 인스턴스 링크

3.1.3 그래프 구조 정의

하이퍼텍스트 공간은 스키마 그래프와 인스턴스 그래프로 이루어지며 각각은 노드·링크로 구성된다.

[정의5] 하이퍼텍스트 스키마 그래프

스키마 그래프(SG)는 {{SN}, {SL}}로 구성된다.

{SN} : 스키마 노드 집합, {SL} : 스키마 링크 집합

스키마 그래프(SG)는 스키마 노드들과 스키마 링크들로 구성된 단 방향성 그래프이다.

[정의6] 인스턴스 그래프

인스턴스 그래프(IG)는 {{IN}, {IL}}로 구성된다.

{*IN*} : 인스턴스 노드 집합, {*IL*} : 인스턴스 링크 집합

인스턴스 그래프(IG)는 SG상의 인스턴스 노드 집합과 단방향성 인스턴스 링크의 집합으로 구성되는 그래프이다.

3.2 프로세스 서브 모델의 확장

하이퍼텍스트 그래프에서 가장 중요한 정보 검색 수단은 항해 연산으로, 제안 모델에서는 기존의 항해 연산에 기능 모델로 정의된 질의 기능을 추가하여 다양한 항해 검색 유형을 지원할 수 있도록 프로세스 서브모델을 확장한다. 기능 모델을 통해 정의되는 동적 링크 확장은 웹 데이타베이스의 후위 구조와 데이타 독립성을 유지할 수 있는 장점을 지니며,[4] 동적·정적 링크 모

두를 정의 할 수 있고, 완전한 링크의 동적 성질과 확장성, 호환성, 융통성을 지원한다[2].

3.2.1 가상 구조

가상 구조는 실제 존재하는 스키마 구조에 대한 접근 요구 시, 동적으로 생성되는 논리적 구조이다.

[정의7] 가상 노드(VN : Virtual Node)

가상 노드(VN)은 VN_Name(v_slot_1, v_slot_2, ... , v_slot_n, DEF)로 구성된다.

<VN_Name : 가상 노드 이름, v_sloti : DEF 명세로부터 추출된 동적 슬롯>

<DEF : SQL형태의 기능명세 구문>

[제약조건3] ①단일 인스턴스 노드의 내용으로 가상 노드를 구성할 경우 <v_1, v_2, ... , v_m>과 같이 나타내며, 각 슬롯 값 v_i는 인스턴스 노드의 슬롯 값을 참조로 하여 구성한다(단, 1 < i < m). ②복수 인스턴스 노드의 내용으로 가상 노드를 구성하면 <<v_{11}, v_{12}, ... , v_{1n}>, ... ,<v_{m1}, v_{m2}, ... , v_{mn}>> 과 같이 표현된다. 특정 슬롯의 내용에는 <v_{i1}, v_{i2}, ... , v_{in}>과 같이 복수 개 인스턴스 노드의 슬롯 값이 할당되며, 각 v_{ij}는 인스턴스 노드의 슬롯 값을 참조로 하여 구성한다 (단, 1 < i < m, 1 < j < n).

가상 노드(VN)는 항해 연산 수행 시, 인스턴스 구조의 실제 값을 추출하여 동적으로 노드의 내용을 구성한다. 이때, 목표 가상 노드 생성을 위한 포인트 역할을 하는 링크레이블이 동시에 포함될 수 있는데, 링크 레이블에는 질의 명세가 동적으로 구성되어 내포된다. 가상 노드는 <표 3-(1)>과 같이 표기한다.

[정의8] 가상 링크(VN : Virtual Link)

가상 링크 (VL)은 VL_Name(DEF, Cardinality)로 구성된다.

<VL_Name : 가상링크 이름, Cardinality:링크 연관 수>

<DEF : 근원노드와 목표노드의 링크 설정과 관계된 SQL형태의 조건절 명세>

가상 링크는 가상 노드 생성을 위해 술어로 명세된 기능 명세 조건절을 수행하여 인스턴스 구조를 참조하는 논리적 링크이다. 가상 링크는 <표 3-(2)>와 같이 표기한다.

3.2.2 가상 그래프

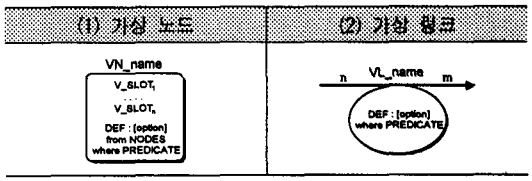
[정의9] 가상 그래프(VG : Virtual Graph)

가상 그래프(VG)는 {{VN}, {VL}}로 구성된다. {VN} : 가상 노드 집합, {VL} : 가상 링크 집합

- 생성 측면에서의 가상 그래프 구조

<그림 6>은 가상 노드와 가상 링크의 생성 과정을 나타낸다. '작가', '작품'은 각각 근원 스키마 노드, 목표

표 3 가상 구조의 표시 기호



스키마 노드를 나타내며, '작품정보'는 두 스키마 노드간의 스키마 링크를 나타낸다. 또한, 작품번호, 작가번호는 식별자 슬롯을, 근원노드, 목표노드는 근원 슬롯, 목표 슬롯을 각각 나타낸다. 근원가상노드와 목표가상노드는 인스턴스 구조를 참조하여 자신의 가상노드 기능명세에 의해 동적으로 생성되며, 가상링크는 근원가상노드와 목표가상노드의 해당 인스턴스를 가상링크 기능명세인 "DEF"절을 참조하여 동적으로 연결된다.

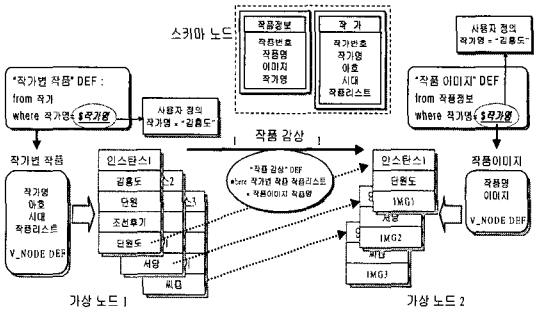


그림 6 가상 링크의 기능 명세

<그림 7>은 본 연구에서 제안한 모델링 기법을 적용한 전자 미술관 모델의 일부분이다.

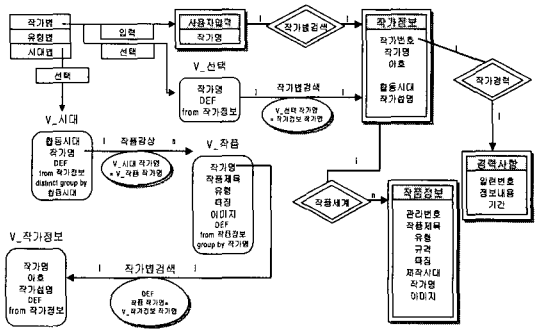


그림 7 전자미술관의 확장된 하이퍼텍스트 모델링 예

3.3 구현 모델(implementation model)

본 절에서는 표현 모델의 확장 개념을 관계형 데이타 모델과 상호 사상한다.

3.3.1 기본 노드 사상 정의

본 절에서는 입력노드에 대한 기본 사상을 다룬다. 입력 노드는 동적 가상 노드에서 식별자 슬롯 즉, 기본키 애트리뷰트 값을 선택하는 역할을 수행하며, 이 경우 자신의 릴레이션과 그 릴레이션에 대해 외래키로 참조 관계에 있는 릴레이션들을 선택하여 테이블로 표현하며, 검색자는 조건 값을 지정한다.

3.3.2 확장 노드 사상 정의

확장된 하이퍼텍스트 모델을 관계형 데이타 모델로 사상하는 사상 기법은 다음과 같은 3단계 과정으로 이루어진다.

(1) 단계 1 - 스키마 노드의 사상

- 스키마 노드 SN에 대해 관계형 스키마 R을 생성하고 스키마 노드 SN의 이름은 관계형 스키마 R의 이름으로 사상한다. 스키마 노드 SN의 각 슬롯은 관계형 스키마 R의 애트리뷰트로 사상하고 스키마 노드 SN의 식별자 슬롯은 관계형 스키마 R의 기본키 애트리뷰트로 사상한다. 이때, 슬롯의 숫자 및 텍스트 타입은 관계형 스키마 R의 도메인으로 사상한다. 기본 사상을 정리하면 <표 4>와 같다.

표 4 스키마 노드의 사상 관계

스키마 노드 구조	관계형 스키마 구조
스키마 노드 SN	릴레이션 R 생성
스키마 노드 SN의 이름	릴레이션 R의 이름
슬롯	애트리뷰트 생성
식별자 슬롯	기본 키
슬롯 타입	애트리뷰트의 도메인

(2) 단계 2 - 스키마 링크의 사상

단계 2에서는 스키마 링크(SL)를 관계형 모델로 사상하며 스키마 링크의 카디널리티에 따라 세분된다.

- **1 : 1 / 1 : N 카디널리티** : 연관된 스키마 노드 SN에 대한 관계형 스키마 R의 외래키로 상대 스키마 노드 SN′ 에 대한 관계형 스키마 R′ 의 기본키 즉, 근원노드 또는 목표노드의 식별자 슬롯을 근원(1:1) 또는 목표(1:N) 릴레이션에 삽입한다.
- **M : N 카디널리티** : 스키마 링크 SL에 대한 새로운 관계형 스키마 R을 생성한다. 스키마 링크 SL의 이름을, 생성된 관계형 스키마 R의 이름으로 사상하고, 스키마 링크의 링크 슬롯은 생성된 관계형 스키마 R의 애트리뷰트로 사상한다. 또한 관계형 스키마 R의 기본키는 독립적인 애트리뷰트 필드를 생성하여 삽입하거나 스키마 링크로부터 사상된 근원 슬롯, 목표 슬롯의 쌍으로 구성한다. <그림 4>의 하이퍼텍스트 스키마 구조를 사상 알고리즘에 따라 관계형 스키마로 사상하면 <그림 8>과 같다.

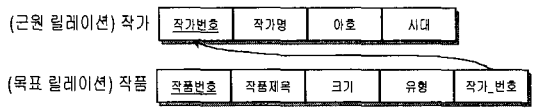


그림 8 관계형 스키마로 사상 예

(3) 단계 3 - 가상 그래프의 사상

- **가상 노드**는 'DEF' 명세를 포함한 **SQL 뷰**로, 가상 노드의 슬롯은 애트리뷰트로 사상한다.
- **가상 링크**는 검색 수행 시에 'DEF' 명세 조건을 만족하는 **SQL질의**를 생성하여 사상한다.

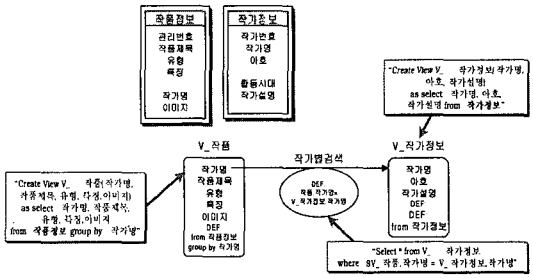


그림 9 가상 그래프의 사상 기법 적용 예

<그림 9>는 가상 노드 "V_작품", "V_작가정보"와 이 두 노드간의 가상 링크 "작가별검색"으로 이루어진 가상 그래프의 예이다. 가상 노드들은 <그림 9> 상단에 표시한 스키마 노드 "작품정보", "작가정보"를 기반으로 정적으로 존재하는 정보로부터 동적 질의에 의해 생성된다. 이때, 가상 노드는 "DEF" 명세에 의한 **SQL 뷰**로 생성되며, 가상 링크 "작가별검색"은 두 가상 노드들의 인스턴스를 동적으로 연결할 수 있도록 **SQL 질의**로 변환되어 링크 항해 연산 시에 호출된다.

3.3.3 항해 유형 분류 및 노드·링크 생성 규칙

본 절에서는 가상 그래프 공간 내에서 검색 연산에 의해 생성 가능한 항해 유형을 분류하여 동적인 간접

질의 발생 형태를 정의하며, 이를 바탕으로 하이퍼텍스트 모델과 관계형 데이타 모델간의 노드·링크 생성 규칙을 정의한다. 사상 관련 기호는 <표 5>와 같다.

표 5 사상 관련 기호

기 호	의 미
Vmn	VN의 m번째 슬롯의 n번째 인스턴스 값
Lm	VN의 m번째 링크 레이블
GN	기존의 정적 성질의 일반 노드
IN_Node	입력 노드
Ri	i번째 릴레이션(테이블)
==>	노드, 링크 생성 방향
.(dot)	부속 관계
|(or)	선택 사항
[a, b]	a and/or b 의 원소를 갖는 리스트
Ri[Aj]	i번째 릴레이션의 j번째 애트리뷰트
AVij	질의 결과 릴레이션의 i번째 튜플에서 j번째 애트리뷰트 값

<그림 10>은 가상 그래프 공간에서 항해 유형을 분류한 것이며, 항해 유형 각각에 대한 가상 노드·가상 링크의 생성 규칙을 <표 5>의 기호를 사용하여 나타내면 <표 6>과 같다.

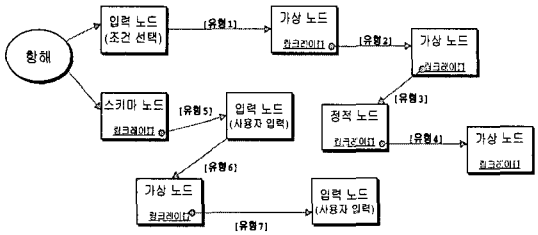


그림 10 항해 유형 분류

4. 웹 데이타베이스 연동 기법

CGI 서버 방식에서는 여러개의 질의처리 서버가 존재하여도 특정 서버에 병목현상이 발생되며, 이로 인해 데이타베이스 접근 동시성이 저하되는 문제점을 가지고 있다. 따라서 본 논문에서는 질의처리 요구 유형에 가중치를 할당하고, 할당된 가중치와 질의처리 서버의 현재 부하량을 기반으로 동적 서버 할당 스케줄링 기법을 제안한다. 또한 스케줄링의 유연성을 향상시키기 위해 질의처리 요구 수행 후의 질의 응답 시간을 가중치 변화에 적용하는 가중치 재산정 기법도 제안한다. <그림 11>은 제안기법을 적용하기 위해 확장된 데이타베이스 통로의 논리적 구성도이다.

표 6 가상 노드·링크 생성 규칙

유형	노드	사상 규칙
1	입력=>가상	IN_Node=(조건선택/입력)=>[AV_{11},...,AV_{mn} ,L_1,...,L_m]
2	가상=>가상	[AV_{11},...,AV_{mn}, L_1,...,L_m]=(링크레이블선택)=> [AV_{11},...,AV_{mn} ,L_1,...,L_m]
3	가상=>정적	[AV_{11},...,AV_{mn}, L_1,...,L_m]=(링크레이블선택)=>GN
4	정적=>가상	GN=(링크레이블선택)=>[AV_{11},...,AV_{mn}, L_1,...,L_m]
5	스키마=>입력	[R_1,R_2,...,R_i]=(릴레이션선택)=>[R_1[A_1,...,A_j],... R_i[A_1,...,A_j]]
6	입력=>가상	[R_1[A_1,..,A_j],..,R_i[A_1,...,A_j]]=(조건선택)=> [AV_{11},...,AV_{mn}, L_1,...,L_m]
7	가상=>입력	[AV_{11},...,AV_{mn}, L_1,...,L_m]=(식별자슬롯선택)=> R_1[A_1,..,A_i],..R_i[A_1,..,A_j]]

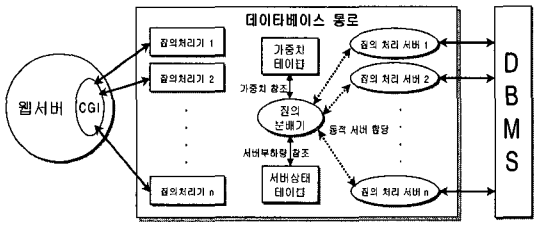


그림 11 확장된 동적 데이타베이스 통로

4.1 응답 시간 예측을 통한 질의 분류

웹 데이타베이스 시스템 환경에서는 서비스 수행에 필요한 질의를 모델링 시에 산출 가능하고, 이를 바탕으로 질의의 일반적인 응답 시간을 예측할 수 있다. 이러한 특수성을 이용하여 조인 연산의 유무(2-way 조인만 고려), 셀렉션의 유무, 범위 탐색의 여부, 키의 사용 여부 등을 고려하여[18] Q1에서 Q7까지 질의 유형을 분류하였다. <그림 12>는 질의 유형 분류에 관계된 고려 대상과 분류된 질의 유형을 계층적으로 표현한 것이다.

4.2 초기 가중치 할당

질의 수행 비용은 보조 기억장치의 접근 횟수와, 질의 처리 중간 결과 저장비용, 메모리에서의 연산 비용, 질의 전송과 검색 결과 전송에 소요되는 통신비용을 기준으로[5][18]다음과 같이 요약한다. <수식 1>을 적용한 결과를 질의 유형별 초기 가중치로 할당하여 질의처

리 서버 할당 스케줄링의 참조 값으로 활용한다

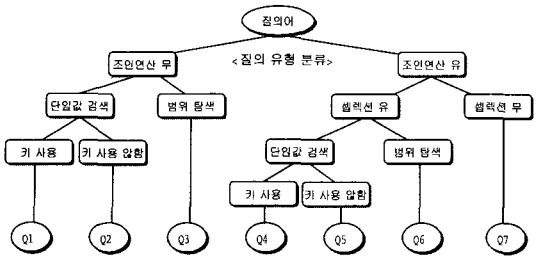


그림 12 질의 유형 분류

X_A : 애트리뷰트 A의 인덱스 레벨의 수
S_A : 애트리뷰트 A의 셀렉션 카디널리티
bfr : 블러킹 인수
B : 전체 블럭 수
$|T_1|$: 테이블 T_1의 레코드수
Js : 조인연산 선택

$$W_1 = X_A + 1$$

$$W_2 = X_A + \lfloor \left(\frac{S_A}{bfr}\right) \rfloor$$

$$W_3 = X_A + \frac{B}{2}$$

$$W_4 = (X_A + 1) + (X_B + \lfloor \frac{S_B}{bfr} \rfloor) + \frac{(Js|T_1||T_2|)}{bfr}$$

$$W_5 = (X_A + \lfloor \frac{S_A}{bfr} \rfloor) + ((S_A(X_B + 1)) + \frac{(Js|T_1||T_2|)}{bfr}$$

$$W_6 = (X_A + 1) + (\frac{|T_1|}{2}(X_B + 1)) + \frac{(Js|T_1||T_2|)}{bfr}$$

$$W_7 = BT_1 + |T_1|(X_B + 1) + \frac{(Js|T_1||T_2|)}{bfr}$$

수식 1 초기 가중치 산정 계산식

4.3 가중치 재산정 기법

웹 데이타베이스 시스템의 서비스 응답 시간은 동작 환경에 따라 다양한 결과를 나타내기 때문에 시스템 설계 단계에서 이를 정확히 예측하기가 어렵다[17]. 따라서 본 절에서는 시스템 동작 환경에 유연하게 적응 가능한 가중치 재산정 기법을 통하여 처리된 질의의 평균 응답 시간을 기존 가중치에 반영하고, 이를 질의처리 서버 할당 시에 적용할 수 있도록 한다. <그림 13>은 가중치 재산정 과정을 도식화 한 것이다.

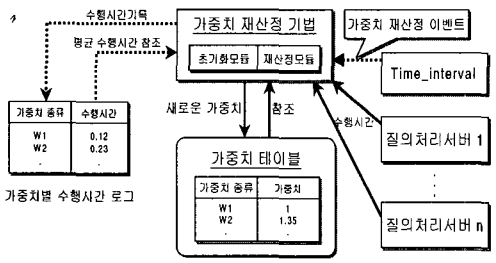


그림 13 가중치 재산정 과정

```
/* time_interval > EVALUATE_TIME_INTERVAL 일때 */
set_evaluate_time();                    /* 재산정 시작 시간 기록 */
while( there exists any log entry){     /* Log file에서 수행시간 읽기 */
    read(log_file, weight_no, exe_time);
    weight_cnt[weight_no] ++;
    weight_tot[weight_no] += exe_time;
}
for( no =1; no <= WEIGHT_NO; no ++){    /* 가중치별 임시 가중치 */
    if (weight_cnt[no] > 0) {
        weight_ave[no] = weight_tot[no] / weight_cnt[no];
        Temp_weight[no] = weight_ave[no] / weight_ave[1];
    }
}
Lock Weight_Table;                      /* 가중치 테이블에 Lock */
for( no =1; no < WEIGHT_NO; no ++){
    if (Temp_weight[no] > 0) {
        New_weight[no] = AVG(Weight_Table[no], Temp_weight[no]);
        Weight_Table[no] = New_weight[no] ;/* 새로운 가중치 반영 */
    }
}
UnLock Weight_Table;                    /* 가중치 테이블에 UnLock */
delete all exe_time_log_records;        /* Log file을 지운다 */
```

질의 처리 서버는 질의 수행 후, 질의 결과와 함께 서버에서의 질의 수행 시간을 가중치별 수행 시간 로그에 기록하고, 기록된 가중치별 질의 수행 시간 로그를 참조하여, 각 가중치별 평균 수행 시간을 기준으로 임시 가중치를 산정한다. 수행 중에는 임시 가중치와 기존 가중치의 평균을 구하여 질의 수행의 실제 응답시간이 가중치 테이블에 반영되도록 한다.

4.4 질의 처리 서버 스케줄링 기법

질의처리 서버 스케줄링 기법은 크게 트랜잭션 풀(pool) 관리, 서버 상태관리, 질의처리 서버 할당의 세 부분으로 구성된다. 다중의 질의처리 요구가 도착하면 질의의 가중치와 질의처리 서버의 부하량을 참조하여 질의처리 서버를 동적 할당함으로써 데이타베이스 통로의 병목 현상을 줄여 시스템 성능을 향상시킨다.

서버 상태 관리 과정은 서버 상태 테이블을 생성하고, 서버 상태 값을 초기화하며, 주기적으로 질의 처리 서버에게 보내는 폴링 메시지를 통해 서버의 가용성을 점검한다. 서버 상태 테이블은 질의 처리 서버에 의해 참조되는 자료 구조이며, 현재 처리중이거나 처리를 기다리는 트랜잭션 정보를 포함한다. <그림 14>는 서버 상태 관리 과정이 생성, 갱신, 참조하는 질의처리 서버의 상태 테이블 자료 구조이며, <그림 15>는 질의처리 서버 상태 관리 과정을 도식화한 것이다.

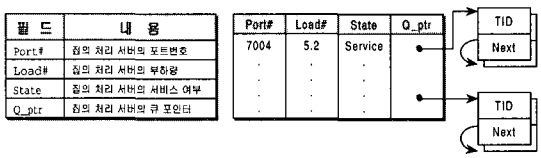


그림 14 서버 상태 테이블의 구조

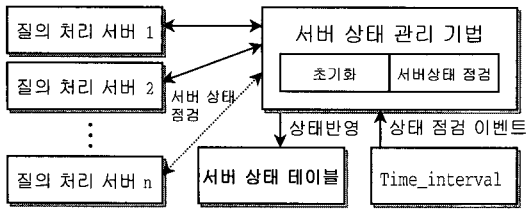


그림 15 서버 상태 관리 과정

질의처리 서버 할당 기법은 트랜잭션 풀에서 가장 작은 가중치를 가진 트랜잭션을 선택하여(LWF : Lowest Weight First), 이를 현재 질의처리가 가능한 질의처리 서버 중 전체 부하량이 가장 작은 서버의 처리대기 큐에 할당한다(MLF : Minimum Load First). 따라서 모든 질의처리 서버가 부하량에 따라 동적으로 할당되어 질의처리 서버의 가용성과 병목현상을 해결 가능하며, DBMS의 동시성을 최대한 지원할 수 있다. <그림 16>은 질의처리 서버 할당 기법을 도식화한 것이다.

```
/**** Lowest Weight First 알고리즘 ****/
  TID = LWF();              /* 트랜잭션 풀에서 가장 작은 가중치를 선택 */
  Server_no = MLF();        /* 가용한 서버중 최소의 부하를 가진 서버 검색 */
  Lock Server_State_Table;  /* 서버 상태 테이블을 Lock */
  Assign_Queue(Server_no, TID);/* 서버 큐에 트랜잭션을 할당 */
  Plus_Server_Load(Server_no, TID_Weight);/* 서버의 부하량을 증가 */
  UnLock Server_State_Table; /* 서버 상태 테이블을 UnLock */
  Lock Transaction_Pool;    /* 트랜잭션 풀을 Lock */
  Transaction_Scheduled(TID);/* 트랜잭션이 스케줄링 되었음을 표시 */
  UnLock Transaction_Pool;  /* 트랜잭션 풀을 UnLock */

/**** Minimum Load First 알고리즘 ****/
while (State_Table_Lock == YES) ;/* 서버 상태 테이블이 Lock이면 대기 */
for(no=1; no <SERVER_CNT; no++)
{  if (Server[no].state == SERVICE && Server[no].load < Minimum_Load)
     { Minimum_Load = Server[no].load;
       server_no = no;
     }
}
Lock Server Table ;                  /* 서버 상태 테이블을 Lock */
Assign_Queue (Server[server_no], TID);  /* 서버 큐에 트랜잭션을 할당 */
Plus_Load (Server[server_no], weight);  /* 서버의 부하량(load)을 증가 */
Unlock Server Table ;                /* 서버 상태 테이블을 Unlock */
Transaction_Scheduled(TID);          /* 스케줄링 되었음을 표시 */
```

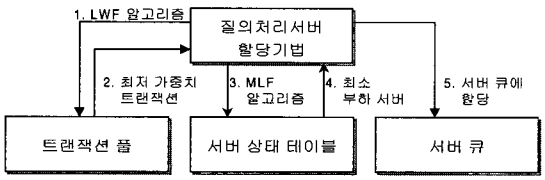


그림 16 질의처리 서버 할당 기법

5. 성능 평가

본 논문에서 제안한 데이타베이스 통로를 <표 7>과 같은 환경에서, 질의에 따라 정적으로 질의처리 서버가 고정되는 기존의 CGI 서버 방식과 <그림 19>와 같은 구조로 비교 평가하였다. 또한 본 연구에서 제안한 스케줄링 기법과 질의처리 서버를 무작위로 할당하는 임의 서버할당 기법과도 비교 실험하였다.

표 7 성능 평가 환경

시스템 구현 환경	
서버	Sun Sparc 20 (운영체제 : Solaris 2.4)
클라이언트	Axill 220 (운영체제 : Sun OS 4.1.3)
DBMS	Informix 7.2
웹 서버	NCSA 1.5.2
컴파일러	SC3.0, gcc 2.7.2
기타	TCP/IP Module, Perl 5.0

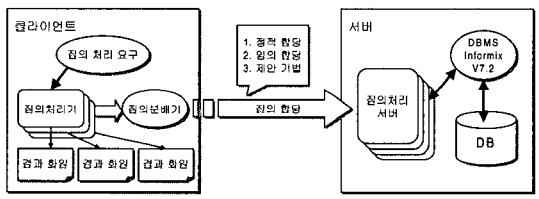


그림 17 성능 평가 구조

먼저 1단계 실험에서는 질의 처리 서버를 2개로 고정시킨 후, Q1(단순질의), Q2(복잡질의) 형태의 두 가지 유형 샘플 질의를 무작위로 15개, 31개를 동시 생성하여 기존 CGI서버 기법인 정적 서버 할당과 임의 서버 할당 기법을 평가 실험하였다. <그림 18>은 성능 평가에 사용된 질의 유형 예와 실험용 스키마를 나타낸 것이다.

실험 결과를 통해 동시발생 질의 처리 요구가 늘어나고 질의 유형이 복잡해질수록 스케줄링의 필요성이 증가함을 알 수 있다.

질의 유형	질의 복잡도	연산 유형	결과 크기
Q1	단순	판독	1개 튜플
SELECT name, addr from artist WHERE num='1';			
Q2	복잡	판독	n개 튜플
SELECT name, addr from artist, work_info WHERE num='1' or num='2';			

```
CREATE artist
            num         CHAR(20) UNIQUE NOT NULL;
            name        CHAR(30);
            addr        CHAR(30);
;
CREATE INDEX ON artist(    num  );

CREATE work_info
            no          CHAR(20) UNIQUE NOT NULL;
            title       CHAR(30);
            wno         CHAR(30);
;
CREATE INDEX ON work_info(no);
```

그림 18 성능 평가 실험용 질의 유형의 예와 스키마

2단계 실험에서는 평가 질의의 유형을 복잡도를 고려하여 4가지를 생성하여 사용하였으며, 질의 처리 서버를

4개로 고정시키고 동시 발생 질의처리 요구를 3, 7, 15, 31, 63개로 변화시키면서 기존의 CGI 서버 방식인 정적 서버 할당과 제안한 동적 스케줄링 기법을 비교 측정하였다. <표 8>은 평가용 질의 유형에 대한 평균 응답 시간이며 <그림 20><그림 21>은 기존의 정적 서버 할당 기법과 제안 기법간의 질의 처리요구 수에 따른 응답 시간 변화를 그래프로 나타낸 것이다.

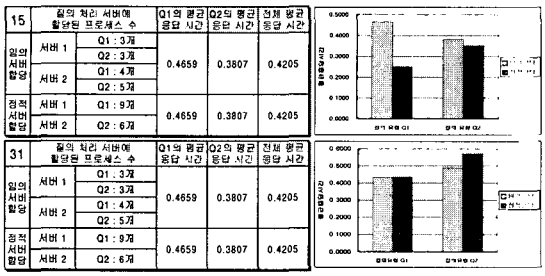


그림 19 정적 할당 기법과 임의 할당 기법 성능 비교

표 8 성능 평가 실험용 4가지 질의유형의 평균응답시간

질의복잡도	평균 질의처리 시간
복잡도 C1	0.004238 초
복잡도 C2	0.113451 초
복잡도 C3	1.105645 초
복잡도 C4	10.046650 초

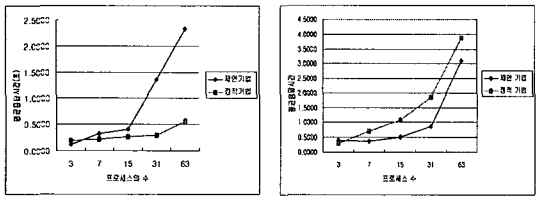


그림 20 질의 복잡도 C1, C2에 대한 평균 응답 시간

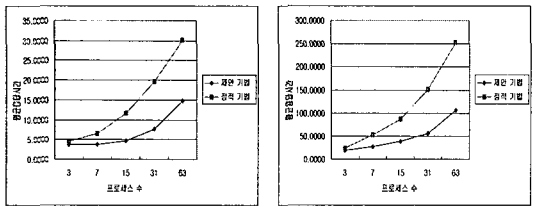


그림 21 질의 복잡도 C3, C4에 대한 평균 응답 시간

실험 결과를 통해, 매우 단순한 질의 복잡도 C1을 제외하고 C2, C3, C4에 대한 실험에서 동시 발생 프로세스 수가 증가함에 따라 기존 기법과 제안 기법간의 응답시간 차이가 최고 2배 이상 나타나는 것을 볼 수 있다. 결과적으로 동시 발생 질의처리 요구가 증가하고 특정 유형의 질의가 집중되어 발생하며, 질의 복잡도가 높은 질의 유형의 요구가 증가할수록 기존의 정적 할당 방식 보다 제안된 기법이 평균 응답 시간 측면에서 우수함을 알 수 있다.

프로세스 수	C2	C3	C4
3	-27.6%	14.4%	23.5%
7	44.3%	42.9%	49.0%
15	53.1%	60.0%	55.1%
31	53.5%	61.1%	62.3%
63	20.2%	50.8%	58.2%

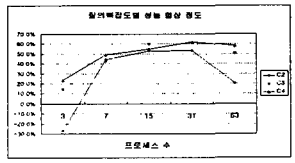


그림 22 질의 복잡도별 성능 향상 정도

6. 결 론

본 연구에서는 기존의 하이퍼텍스트 모델의 단점을 해결하기 위해 하이퍼텍스트 표현 모델을 확장하였다. 데이타 서브모델에서는 기존의 정적인 노드의 개념을 논리적 노드나 그 명세 정보들로 구성된 집합으로 확장하였고, 링크도 물리적인 링크 개념을 논리적 링크나 그 명세 정보들로 구성된 집합으로 확장하였다. 또한 하이퍼텍스트 스키마 그래프, 인스턴스 그래프를 정의하였다. 프로세스 서브모델에서는 기능 모델을 적용하여 내용 탐색을 기반으로 질의에 의한 항해 개념을 지원하도록 확장된 데이타 서브모델로부터 생성되는 가상 구조인 가상 노드와 가상 링크, 실제 가시적인 브라우징 영역인 가상 그래프 등을 정의하였다. 또한 확장된 모델을 관계형 데이타 모델로 사상하기 위한 알고리즘과, 생성된 가상 브라우징 그래프에서의 항해 유형을 분류하고 각각에 대해 노드·링크 생성 규칙을 유도하였다.

데이타베이스 연동 연구에서는 CGI 응용 서버 방식의 질의처리 서버 병목 현상을 해결하기 위하여, 질의 유형 분류, 가중치 할당 기법, 가중치를 기반으로 한 질의처리 서버 스케줄링 기법을 제안하였다. 또한, 제안된 기법과 기존의 정적서버할당 기법을 비교 실험하여 제안된 기법이 유사한 유형의 질의가 동시에 많이 발생하거나 복잡한 질의의 발생 빈도가 증가할수록 정적 할당 방식 보다 우수함을 보였다.

참 고 문 헌

[1] Athman B. et. al., "World wide Database - Integrrating the Web, CORBA and Databased," Proc. of ACM SIGMOD record, volume 28, Issue 2, June 1-3, 1999

[2] Helen ASHMAN Janet VERBYLA, Hypermedia Management in Large-scale Information System Using The Functional Model of the link, Proceedings of 5th Australasian Database Conference '94, 1994, pp 247-257.

[3] David, K. and Oded, S., "W3QS : A Query system for the World-Wide Web," Proc. of the 21st VLDB Conf., 1995

[4] H.L.Ashman and J.L.M. Verbyla, Dynamic Link Management via the Functional Model of the Link, Proc. of the Basque International Workshop on Information Technology, February 1994, pp 327-334.

[5] P. G. Selinger et. al. , "Access Path Selection in a Relational Database Management System," Readings in Database System 2nd , 1994, pp 84-95

[6] Samarati, P. et. al., "An Autorization Model for a Distributed Hypertext System," IEEE Trans. on Knowledge and Data Eng., 8,4, Aug., 1996

[7] D. Florescu, A.Y. Levy, and A.O. Mendelzon, "Database techniques for the world wide web : survey," SIGMOD record, 27(3), 1998

[8] P. C. Kim, "A Taxonomy on the Architecture of Database Gateways for the Web," submitted for review, 1996

[9] D. Robinson. The WWW Common Gateway Interface Version1.1, Internet Draft . Jan, 1996.

[10] M. F. Arlitt and C. L. Williamson, "Web Server Workload Characterization : The Search for Invariants," Proc. of SIGMETRICS96, ACM, Philadelphia, May, 1996

[11] Hara, Y. and et al, "Hypermedia Database "Hitoki", and Its Applicaions," 22th Int'l Conf. on Data Eng., 1996

[12] A.O. Mendelzon, G.A. Mihaila, and T. Milo, "Querying the World Wide Web," Internation Journal on Digital Libraries, 1(1), 1997

[13] Tanaka, K. et. al., "Query Pairs As Hypertext Links," 7th Data Engineering, 1991

[14] Deng, Y. and Chang, S-K., "A HyperNet Model for Large Hypertext Systems," DASFAA, Japan, April, 1991

[15] Paul D.B. et. al., "An Extensible Data Model for Hyperdocuments," Proc., of the 4th ACM Conf. on Hypertext, Models, 1992

[16] P. Atzeni, G. Mecca, and P. Merialdo, "To Weave th Web," Conf. on VLDB'97 August 26-29, 1997

[17] Yew-Huey Liu, Paul Dantzig, C. Eric Wu, Jim Challenger, "A Distributed Web Server and Its performance Analysis on Multiple Platforms" IEEE Proc. of the 16th ICDCS, 1996

[18] Elmasri, R. and Navathe, S.B., Fundamentals of Database Systems, 2nd Ed., The Benjamin/ Cummings Pub., Co., Inc., 1994

신 판 섭

1992년 2월 홍익대학교 전자계산학과 (이학사). 1994년 2월 홍익대학교 대학원 전자계산학과 (이학석사). 1994년 3월 ~ 현재 홍익대학교 대학원 전자계산학과 박사과정. 관심분야는 웹 데이타베이스, 멀티미디어 시스템, 분산병렬 시스템, 클러스터 시스템

김 성 완

1996년 2월 명지대학교 전자계산학과 (공학사). 1998년 2월 홍익대학교 대학원 전자계산학과 (이학석사). 1998년 3월 ~ 현재 홍익대학교 대학원 전자계산학과 박사과정. 1999년 3월 ~ 현재 삼육의명대학 전산정보과 전임. 관심분야는 데이타베이스, 멀티미디어 데이타 모델링, 하이퍼미디어, 웹 응용서비스

임 해 철

1976년 서울대학교 계산통계학과 이학사. 1978년 한국과학기술원 전산학과 이학석사. 1988년 서울대학교 컴퓨터공학과 공학박사. 1989년 ~ 1990년 University of Florida Post doc. 1981년 ~ 현재 홍익대학교 컴퓨터공학과 교수. 관심분야는 멀티미디어, 분산, 객체지향 데이타베이스