RFID 태그의 색인을 위한 위치 식별자 재순서 기법

(Reordering Scheme of Location Identifiers for Indexing RFID Tags)

안 성 우 † 홍 봉 희 ††
(Sungwoo Ahn) (Bonghee Hong)

요 약 RFID 태그의 시공간 이력정보는 리더에 의해 수집된 선분인 태그간격으로 모델링될 수 있으며, 태그 식별자(TID), 위치 식별자(LID), 시간(TIME)을 축으로 하는 3차원 도메인에서 색인될 수 있다. 도메인 공간에서 태그간격의 분포는 태그 위치추적 질의의 성능을 결정짓는 주요 요소이며 이는 각 도메인 좌표의 정렬에 따라 달라진다. 특히, 시간에 따라 변경되는 태그의 위치 이력을 검색하는 태그 위치추적 질의는 위치정보를 제공하는 LID가 도메인에서 정렬되는 순서에 따라 성능이 달라진다. 따라서, 색인에 저장된 태그간격의 검색 성능 향상을 위해서는 최적의 LID 순서를 결정하는 것이 필요하다. 이를 위하여 이 논문에서는 LID 간의 새로운 순서화 기준으로써 적용하기 위한 LID 근접성을 정의하고, 질의 시 함께 접근되는 태그간격을 색인에서 근접 저장하기 위한 LID 근접성 함수를 제안한다. 또한, 이를 기반으로 이미 부여된 LID의 재순서 기법을 제안한다. 성능 평가 결과 이 논문에서 제안한 LID 재순서 기법을 색인에 적용했을 때 기존의 LID 부여방식보다 월등한 질의 성능 향상을 보여주고 있다.

키워드 : RFID, 미들웨어, 태그 위치추적 질의, 위치 식별자, 근접성, 순서화, 시공간 데이터베이스

Abstract Trajectories of RFID tags can be modeled as a line, denoted by tag interval, captured by an RFID reader and indexed in a three-dimensional domain, with the axes being the tag identifier (TID), the location identifier (LID), and the time (TIME). Distribution of tag intervals in the domain space is an important factor for efficient processing of a query for tracing tags and is changed according to arranging coordinates of each domain. Particularly, the arrangement of LIDs in the domain has an effect on the performance of queries retrieving the traces of tags as times goes by because it provides the location information of tags. Therefore, it is necessary to determine the optimal ordering of LIDs in order to perform queries efficiently for retrieving tag intervals from the index. To do this, we propose *LID proximity* for reordering previously assigned LIDs to new LIDs and define the LID proximity function for storing tag intervals accessed together closely in index nodes when a query is processed. To determine the sequence of LIDs in the domain, we also propose a *reordering scheme of LIDs* based on LID proximity. Our experiments show that the proposed reordering scheme considerably improves the performance of queries for tracing tag locations comparing with the previous method of assigning LIDs.

Key words : RFID, middleware, query for tracing tags, location identifier, proximity, ordering, spatio-temporal database

· 이 논문은 2009년 교육과학기술부로부터 지원받아 수행된 연구임(지역거점연구단육성사업/차세대물류IT기술연구사업단)

† 학생회원 : 부산대학교 컴퓨터공학과
swan@pusan.ac.kr
†† 종신회원 : 부산대학교 컴퓨터공학과 교수
bhhong@pusan.ac.kr

논문접수 : 2008년 11월 19일
심사완료 : 2009년 3월 10일

정보과학회논문지: 데이타베이스 제36권 제3호(2009.6)

1. 서 론

RFID(Radio Frequency IDentification) 태그를 부착한 객체의 위치를 추적하는 질의는 RFID 적용 시스템에서 가장 중요한 요구사항 중의 하나이다[1-4]. 예를 들어, 그림 1과 같이 농산물의 출하 시 RFID 태그를 부착함으로써 생산지에서 소비자까지의 각 단계에서 발생한 정보를 실시간으로 수집할 수 있으며 농산물의 유통 정보를 조회하는 것이 가능하다. 만약, 소비자가 구매한 농산물이 문제가 발생하여 역학 조사가 필요할 경우 RFID 태그를 이용하여 수집된 정보를 검색함으로써 생산지 확인, 함께 유통된 농산물 추적 및 폐기 등의 조치를 신속히 할 수 있다. 이를 위해서, RFID 기술에 대한 사실상 표준을 담당하고 있는 EPCglobal[5]의 EPCIS(EPC Information System)[6,7]에서는 태그를 부착한 제품의 추적에 대한 다양한 요구사항과 그에 따른 예제 시나리오를 제시하고 있다. 이와 같은 응용에서는 시간에 따른 물품의 연속적인 이동에 의해서 대량의 시공간 이력데이터가 발생하며 이를 효율적으로 추적하기 위해서는 태그의 시공간 이력데이터에 대해서 색인을 하는 것이 필요하다.

태그는 특정 리더의 인식 영역에 들어가고 나올 때 리더는 해당 태그에 대해서 각각 *Enter* 이벤트와 *Leave* 이벤트를 발생시킨다[8]. 각 이벤트는 인식된 태그의 식별자(Tag IDentifier: TID), 태그를 인식한 리더의 위치 식별자(Location IDentifier: LID), 태그를 인식한 시간(TIME)에 대한 정보를 가지고 있으며, 태그 궤적은 *Enter* 이벤트와 *Leave* 이벤트를 통해서 수집된 두 개의 시공간 위치정보를 연결한 선분인 태그간격(Tag Interval: TI)으로 모델링 할 수 있다. 또한, 태그의 위치를 추적하기 위해서 수집된 태그간격은 TID, LID, TIME을 축으로 하는 3차원 도메인에서 색인될 수 있다. 질의 처리기는 태그간격을 저장하고 있는 색인에서 3차원 도메인의 각 조건자를 만족하는 정보를 검색할 수 있어야 한다.

데이터 공간에서 태그간격의 분포는 색인에서 노드의 구성을 결정하며 태그 위치추적 질의의 성능에 영향을 미친다. 즉, 질의 시 함께 자주 접근되는 태그간격이 데이터 공간에서 근접 분포한다면 해당 태그간격은 색인에서도 근접한 노드에 저장되어 질의 처리기가 최소한의 노드 접근으로 결과를 얻을 수 있다. 기존의 이동체 데이터베이스[9,10]에서는 이를 위하여 공간 및 시간 도메인으로 구성된 3차원 색인을 사용하고 있다. 색인을 구성하는 각 도메인은 이동체 궤적의 공간 및 시간 근접성을 반영하며, 이러한 근접성은 질의 시 함께 접근되는 이동체 궤적이 색인의 노드에서 근접해서 분포하도록 한다. 마찬가지로, 태그 위치추적 질의의 검색 성능을 향상시키기 위해서는 태그간격 색인의 각 도메인이 태그간격 간의 근접성을 고려하여 분포될 수 있도록 구성되어야 한다.

태그간격 색인의 도메인 중 LID 도메인은 태그의 심볼릭 위치정보를 제공한다. RFID 시스템에서는 태그가 이동하는 위치에 LID를 부여하기 위하여 리더의 제조사 시리얼 번호(manufacturing scheme), RFID 응용에서 사전적으로 부여한 번호(lexicographic scheme) 등 다양한 방법을 적용할 수 있다. 그러나, 기존의 방식은 태그 위치추적 질의 시 함께 접근되는 태그간격의 근접 저장에 대해서 고려를 하지 않는 문제가 있다. 이로 인해 태그간격의 분포는 도메인에 정렬되는 LID의 순서에 따라 결정되며, 질의 시 함께 접근되는 태그간격이 색인에서 랜덤하게 저장되어 검색 성능을 떨어뜨리게 된다. 따라서, 이를 해결하기 위해서는 이미 부여된 LID를 재정렬하는 방법이 제시되어야 한다.

이 논문에서는 이러한 문제를 해결하기 위해서 LID 근접성 함수(*LID proximity function*)를 정의하며, 이를 기반으로 한 LID 재순서 기법(*reordering scheme*

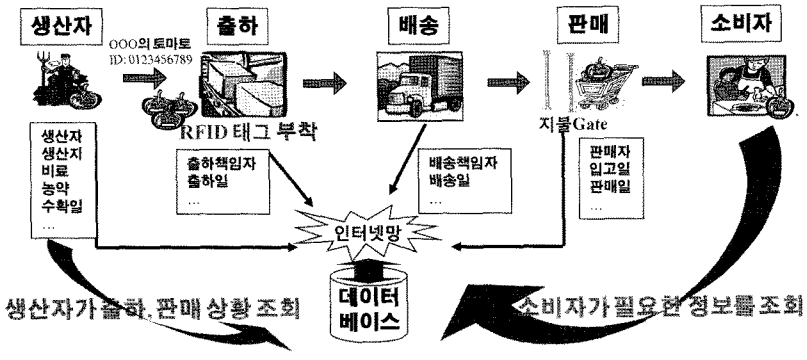


그림 1 RFID를 적용한 농산물 유통 시스템

of LIDs)을 제안한다. 기존에 부여된 LID를 재정렬하기 위하여 먼저 RFID 환경에서 태그를 부착한 물품의 이동에 따라 생성되는 흐름경로를 분석한다. 이를 기반으로 LID 간의 근접성을 계산하기 위하여 정적 요소와 동적 요소로 구성된 LID 근접성 함수를 제시한다. 또한, 색인의 도메인에 적용할 LID 순서를 결정하기 위하여 LID 근접성을 기반으로 한 가중치 그래프를 구성하며 순서화된 정점 집합을 생성함으로써 각 정점에 대응하는 LID 순서 리스트를 생성한다. 마지막으로 이 논문에서 제안하는 LID 근접성을 기반으로 재정렬한 LID와 기존의 방법으로 부여한 LID를 도메인으로 사용하는 색인의 성능비교를 통하여 LID 재순서 기법의 우수성을 입증한다.

이 논문의 구성은 다음과 같다. 2장에서는 공간 데이터베이스에서 객체의 클러스터링에 관련된 기존의 연구에 대해서 분석을 한다. 3장에서는 태그의 위치정보 제공자로써의 LID의 문제점 및 LID 재순서화의 필요성에 대해서 설명을 하며, 4장에서는 RFID 시스템을 구성하고 있는 위치 및 태그 이동의 특징을 기반으로 한 흐름경로를 분석하며, 이를 기반으로 LID 근접성 함수를 정의한다. 5장에서는 4장에서 정의된 흐름경로 및 LID 근접성 함수를 기반으로 구성된 가중치 그래프를 이용한 LID 재순서 기법을 제안한다. 6장에서는 제안한 LID 재순서 기법에 대한 성능 평가를 수행하며, 마지막으로 7장에서는 결론 및 향후 연구를 기술한다.

2. 관련 연구

2.1 EPCglobal 아키텍처 프레임워크

RFID 기술에 대한 사실상 표준을 담당하고 있는 EPCglobal[5]은 RFID 정보에 대한 글로벌 네트워크 구축을 목표로 다양한 표준 명세를 제안하고 있다. EPCglobal에서 표준으로 제시한 EPCglobal 아키텍처 프레임워크(EPCglobal Architecture Framework)[11]는 RFID 태그를 이용하여 RFID를 적용한 물류 환경의 개선 및 발전을 위한 핵심 서비스와 그와 관련된 하드웨어 및 소프트웨어 전반에 걸친 표준에 대해서 기술한다.

EPCglobal 아키텍처 프레임워크는 그림 2와 같이 계층(layer) 구조로 이루어져 있다. 초기에 제시한 구조에서는 시스템의 구현을 중심으로 각 요소를 명세했으나 현재는 네트워크의 구성요소를 하드웨어/소프트웨어 역할(role)과 인터페이스(interface)로 구분을 하고 있다. 이를 기반으로 역할을 담당하는 각 계층은 상위 계층으로 표준 인터페이스를 제공한다. 프레임워크의 가장 하층부분은 태그와 리더가 있으며 둘 사이의 통신을 정의하는 태그 인터페이스(Tag Air Interface)[12]가 있다. 리더에서 수집된 태그 정보는 리더 인터페이스(Reader

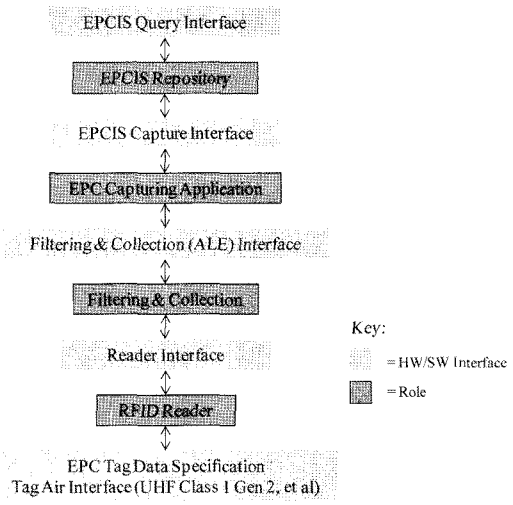


그림 2 EPCglobal 아키텍처 프레임워크

Interface)[13]를 통해서 ALE(Application Level Events)[14]로 전달되며 ALE는 태그 데이터에 대한 여과 및 수집 과정을 수행하며 여과 및 수집 인터페이스(Filtering & Collection Interface)를 통해서 상위 응용에서 요구하는 태그 데이터를 전달한다. EPC 획득 응용(EPC Capturing Application)은 ALE로부터 전달받은 태그 데이터를 획득 인터페이스(EPCIS Capture Interface)를 통해서 EPCIS(EPC Information Service) 저장소[7]에 저장되며 EPCIS 질의 인터페이스(EPCIS Query Interface)를 통해서 태그 데이터가 애플리케이션으로 전달된다.

이 중 ALE와 EPCIS는 태그 데이터의 수집 및 저장을 위한 RFID 미들웨어를 담당하고 있다. 대용량의 태그 데이터에 대해 원활한 서비스를 제공하기 위해서는 미들웨어의 역할이 중요하다. 특히, EPCIS는 다양한 애플리케이션에 태그 데이터를 전달해야 하므로 애플리케이션의 비즈니스 환경에 대응되는 정보를 생성할 수 있어야 한다. 이를 위하여 EPCIS는 ALE를 통해서 수집된 태그 데이터를 다양한 이벤트 형식으로 저장하고 있다. 모든 이벤트는 EPCISEvent로부터 상속되며 EPC로 식별되는 하나 이상의 물리적인 객체의 정보를 포함하는 ObjectEvent, 태그를 부착한 객체 간의 포함 관계를 표현하기 위한 AggregationEvent, 각 EPC 클래스에 해당하는 객체의 수량을 표현하기 위한 QuantityEvent, 대상 RFID 응용에서 정의한 비즈니스 트랜잭션과의 연결 및 분리 정보를 표현하는 TransactionEvent로 구분된다.

2.2 공간 데이터베이스에서의 클러스터링에 관한 연구

기존의 공간 데이터베이스에서는 페이지 클러스터링(object clustering)[15-17]과 페이지 순서화(page order-

ing)[18-20] 기법을 이용하여 질의 시 함께 접근되는 공간 객체를 클러스터링하여 디스크 접근 비용을 줄이고자 하였다. RFID 태그 색인을 위한 LID 재순서화 또한 도메인을 구성하는 값을 정렬하여 함께 접근되는 태그 간격을 색인 노드에서 근접 배치하는 것이 목적이므로 기존의 페이지 클러스터링 및 페이지 순서화와 유사한 성질을 가지고 있다.

페이지 클러스터링은 단일 또는 다중 페이지에 공간적으로 인접한 객체를 저장함으로써 질의 시 디스크 페이지에 대한 접근 횟수를 줄이고자 한다. 단일 페이지 클러스터링 기법[16,17]은 공간적으로 인접한 객체 집합을 동일한 페이지에 모아서 저장하는 방법이다. 질의 시 빈번히 함께 접근되는 공간 객체들이 동일한 페이지에 많이 포함되어 있다면 접근되는 페이지의 수를 줄임으로써 디스크 접근 비용을 줄일 수 있다. 단일 페이지 클러스터링 기법은 페이지 단위로 디스크 입출력이 이루어지는 환경에서 좋은 성능을 발휘하며, 기존의 대부분의 공간 접근 메소드는 단일 페이지 클러스터링에 기반하고 있다. 단일 페이지 클러스터링 기법과 달리 다중 페이지 클러스터링 기법[15]은 물리적으로 인접한 페이지 집합에 공간적으로 인접한 객체의 집합을 분산하여 저장하는 방법이다. 여러 개의 페이지로 구성된 클러스터 단위로 디스크 입출력이 이루어지는 환경에서 한 번의 디스크 접근으로 물리적으로 연속된 페이지를 읽음으로써 공간적으로 인접한 객체들을 가져올 수 있기 때문에 디스크 접근 비용을 줄일 수 있다. 그러나, 클러스터 단위로 디스크 입출력을 하기 때문에 질의 시 요청되지 않는 페이지도 함께 읽기 때문에 추가적인 데이터 전송 비용이 발생하는 문제점이 있다.

페이지 순서화는 공간적으로 인접한 2차원 상의 페이지 또는 공간 객체를 1차원 디스크에 인접하게 저장하기 위한 페이지 순서를 결정하여 질의 시 디스크 접근 횟수를 줄이기 위한 방법이다. 지금까지 대부분의 페이지 순서화에 대한 연구는 특정한 공간 채우기 곡선(space-filling curve)을 기반으로 진행되었다. [20]은 정적인 공간 데이터베이스 환경에서 디스크 탐색 비용을 줄이기 위한 연구로써 Z-순서화, Gray-code 기반 순서화, Hilbert 곡선 기반 순서화와 같은 여러 가지 공간 채우기 곡선에 대하여 분석 및 실험평가를 수행하였다. 이 연구에서는 Hilbert 곡선 기반의 순서화 방법이 다른 공간 채우기 곡선에 비해 우수한 성능을 보이는 것으로 확인되었으며, 이러한 이유에 의해 지금까지 Hilbert 곡선 기반 순서화 방법이 가장 널리 사용되었다. 그러나, Hilbert 곡선을 포함한 공간 채우기 기반 페이지 순서화 방법은 공간 데이터의 분포에 대한 고려를 하지 않기 때문에 항상 고정된 순서가 결정되고 비규등 분포를 가진 공간 객체에 대해서 곡선 상의 인접한 두 페이지가 공간적으로 인접하지 않는 문제가 발생한다. [18]은 Hilbert 곡선 기반의 순서화 방법의 이러한 문제점을 해결하기 위하여 고정적인 순서화 공간 채우기 곡선 대신에 공간 데이터의 분포와 질의 유형을 고려한 비용 모델을 기반으로 한 최적 페이지 순서화 방법을 제안하였다.

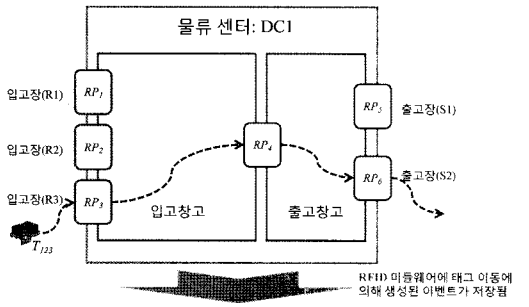
RFID 환경에서 LID 재순서화는 공간 객체를 클러스터링 하기 위한 기존의 연구와 유사성을 가지고 있으나 다음과 같은 이유에 의해 기존의 기법을 그대로 적용하기가 어렵다. 첫째, 페이지 클러스터링과 페이지 순서화 기법은 공간적인 지역성(spatial locality)를 기반으로 객체를 디스크에 저장하고 있다. 즉, 대상 도메인의 좌표 사이에 적합한 근접성이 존재함으로 기본으로 가정하고 있다. 그러나, RFID 환경에서의 위치정보인 LID는 적합한 근접성 정보를 제공하지 못하므로 근접성 정보를 제공해주기 위한 방법이 먼저 제공이 되어야 한다. 둘째, 기존 연구에서 제시하고 있는 공간 객체 또는 페이지의 순서는 객체의 분포에 관계없이 항상 고정되어 있다. RFID 시스템에서 태그는 시간에 따라 자신의 위치를 계속 변경하기 때문에 LID를 이용하여 저장된 태그간격 간의 근접성도 함께 변경된다.

3. 문제 정의

3.1 대상 환경

RFID 환경에서 리더를 통해서 인식된 태그 정보는 RFID 미들웨어에 의해 다양한 이벤트로 표현이 된다. 그림 3은 태그를 부착한 물품이 물류 센터를 통해서 입출고될 때 RFID 미들웨어에서 생성하는 이벤트 정보를 보여주고 있다. 그림 3(a)의 태그 이동 예제에서와 같이 태그가 입고장(receiving dock) R3을 통해서 물류 센터의 창고에 입고된 후 출고장(shipping dock) S2를 통해서 출고되었을 때, 리더가 설치된 각 판독점(Read Point: RP)은 물품에 부착된 태그의 정보를 수집하여 RFID 미들웨어로 전달하며 그림 3(b)와 같은 태그 이벤트 정보가 저장된다. 수집된 태그 이벤트는 태그 식별자(TID), 위치 식별자(LID), 인식시간(TIME)을 포함하고 있으며 태그의 위치를 추적하기 위해서는 TID, LID, TIME을 조건자로 하는 질의처리가 가능해야 한다[6].

태그의 위치정보는 기존의 이동체 데이터베이스에서 사용하는 물리 공간정보와는 달리 심볼릭 위치정보로 표현된다. 그림 3의 예에서 물품이 물류 센터로 들어오기 위해서는 판독점 RP_1, RP_2, RP_3 중 한 지점을 거쳐야 하며, 출고를 위해 RP_4를 통하여 이동하지 않으면 입고 창고에 계속 위치하게 된다. 이에 따라, 태그의 현재 위치는 태그가 인식되었던 지점의 위치정보를 재공



태그	시간	판독점	인식공간
T_{123}	13:00	RP_3	DC1(입고)
T_{123}	15:30	RP_4	DC1(출고)
T_{123}	17:10	RP_6	DC1(이동)

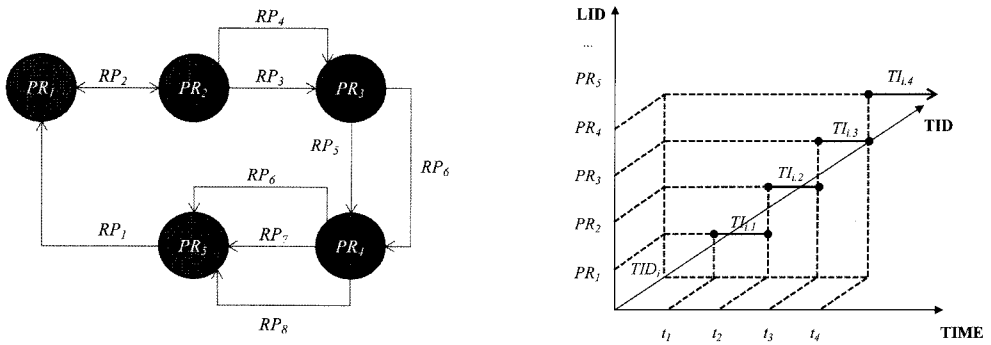
그림 3 RFID 환경에서 태그 이동에 의한 태그 이벤트 생성 예

해주는 판독점 식별자와 판독점 사이에 태그가 머무른 영역의 위치정보를 제공해주는 인식공간(Perceptual Region: PR) 식별자로 표현이 된다.

위치정보 중 판독점 식별자는 태그 이벤트가 발생한 위치를 표현하며 인식공간 식별자는 다음 태그 이벤트가 발생할 때까지 태그가 존재한다고 추정되는 위치를 표현한다. 이는 판독점 단독으로는 태그가 머물렀던 위치에 대한 정보를 제공하지 못한다는 것을 의미한다. 대부분의 RFID 응용은 태그를 부착한 물품의 비즈니스 흐름을 추적하기 때문에 태그의 위치 추적 시 태그가 어디서 입출력 되었는지 보다 어떤 위치에 존재하는지에 대해서 관심을 가진다[7]. 따라서, 태그 위치추적 질의를 위한 LID는 판독점 식별자가 아닌 인식공간 식별자를 사용하는 것이 더 적합하다. 인식공간 식별자는 물리적 리더의 시리얼 번호 할당, RFID 응용의 기준에 따라 미리 정해진 사전적인 번호 할당 등 다양한 방법으로 부여하는 것이 가능하다.

태그 TID_i는 LID_j에 들어가고 나갈 때 태그 이벤트를 발생시키며 입출력 시간을 각각 $TIME_{enter}$, $TIME_{leave}$라고 했을 때 (LID_j, TID_i, $TIME_{enter}$)에서 (LID_j, TID_i, $TIME_{leave}$)를 연결한 선분인 태그간격(Tag Interval: TI)으로써 표현될 수 있다. 이러한 태그간격은 태그가 인식공간을 이동할 때마다 TID, LID, TIME을 축으로 하는 3차원 도메인 상에 모델링 될 수 있으며, 태그 이벤트의 저장을 위해서 색인될 수 있다[8]. 예를 들어, 그림 4(a)와 같이 구성된 공간에서 TID_i가 t_1에서 t_2 시간 사이에 PR_1에 머무르고, t_2에서 t_3 사이에 PR_2, t_3에서 t_4 사이에 PR_3에 머무르고 있다가, t_4에 PR_4에 들어갔다면 그림 4(b)와 같이 TID_i의 태그간격 $TI_{i,1}$, $TI_{i,2}$, $TI_{i,3}$, $TI_{i,4}$가 3차원 도메인에 표현된다. RFID 응용에서 태그를 부착한 물품의 관리를 위해서는 태그간격을 이용하여 물품의 이력 및 현재 위치를 추적하는 것이 필요하다.

태그 위치추적 질의는 조건자와 질의결과에 따라서 표 1과 같이 두 가지의 기본 질의타입으로 분류될 수 있다. 관측 질의(Observation Query: OQ)는 지정된 시간 동안 특정 위치에 머물러 있었던 태그들을 검색하기 위한 질의이며, 궤적 질의(Trajectory Query: TQ)는 지정된 시간 간격 동안 태그가 출입한 위치정보의 이력을 검색하기 위한 질의이다. 태그 위치추적 질의는 조건자에 따라 3차원 공간에서 점(point), 선(line), 평면(plane), 육면체(volume)로 표현된다. 질의 처리기는 데이터 공간에 분포하고 있는 태그간격 중 질의의 영역과 겹치는 태그간격을 검색해야 한다. 이때, 데이터 공간에서 질의의 검색영역을 줄이기 위해서는 태그간격에 대한 색인구조를 구성하는 것이 필요하다.



(a) 판독점과 인식공간 간의 연결정보에 의한 RFID 시스템에서의 위치 표현 (b) 태그 이동에 따른 태그 위치 정보의 3차원 모델링

그림 4 RFID 환경에서 위치의 표현 및 태그 위치정보 모델링 방법

표 1 조건자에 의한 태그 위치추적 질의의 분류

조건자 (Predicate)			질의결과	질의타입
LID	TID	TIME		
point/set/range	*	point/range	TID(s)	관측 질의 (OQ)
*	point/set/range	point/range/*	LID(s)	궤적 질의 (TQ)

3.2 위치 식별자 기반 태그간격 색인의 문제점

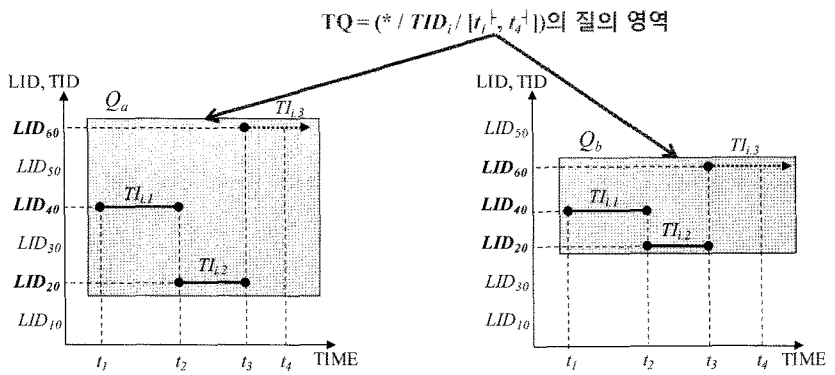
데이터 공간에서 태그간격의 분포는 색인을 통한 질의의 성능을 결정짓는 주요 요소이다. 즉, 질의 처리기가 조건에 맞는 태그간격을 검색할 때 태그간격의 분포는 색인에서 방문해야 하는 노드의 개수를 결정한다. 예를 들어, 질의 Q_a의 검색결과에 포함되는 태그간격 TI_i, TI_j가 데이터 공간에서 가능한 가까이 분포되어있다면 색인은 이 두 태그간격을 같은 노드 또는 이웃 노드에 저장함으로써 색인에서 멀리 떨어져서 저장되는 것보다 노드 탐색 비용을 줄일 수 있다.

태그간격의 분포는 데이터 공간을 구성하는 TID, LID, TIME 도메인의 특성을 따른다. 특히, 태그 위치추적 질의는 태그의 위치 이력을 검색하기 때문에 태그의 위치를 표현하는 LID의 정렬은 질의 시 함께 접근되는 태그간격의 근접 배치에 많은 영향을 미친다. 그림 5는 LID 정렬에 따른 검색영역의 변화를 보여주고 있다. 만약, TID_i가 LID_{20}, LID_{40}, LID_{60}의 순서로 위치를 이동하였다면 도메인에서 LID의 정렬에 따라 그림 5(a), (b)와 같은 태그간격이 생성될 것이다. 그림 5(a)의 경우 LID가 도메인에 식별자의 순서대로 배치가 되어 있기 때문에 $TI_{i,1}$, $TI_{i,2}$, $TI_{i,3}$을 검색하기 위한 궤적 질의 TQ = (* / TID_i / $[t_1^{\vdash}, t_4^{\dashv}]$)는 LID_{20} 부터 LID_{60}를 포함하는 질의영역 Q_a를 설정한다. 그러나, 그림 5(b)와 같이 LID가 TID_i의 이동 순서와 동일한 순서로 도메인에 배치가 되어 있다면 LID_{20}, LID_{40}, LID_{60}만을 포함하는 질의영역 Q_b이 설정되며, Q_a에 비하여 검색영역이 감소하여 색인에서의 노드 탐색 비용을 줄일 수 있다.

그러나, 기존 RFID 시스템에서의 LID 부여방식은 태그 위치추적 질의의 검색영역을 줄이기 위한 태그간격의 분포에 대해서 고려를 하지 않고 있다. 이로 인해 질의 시 함께 접근되는 태그간격은 LID의 정렬에 따라 데이터 공간에서의 분포가 달라지며, 이러한 태그간격은 색인에서 랜덤하게 저장되는 문제가 발생한다. 결과적으로 색인을 통하여 태그간격을 검색할 때 많은 노드의 탐색이 필요하게 되어 노드 접근 비용이 증가한다.

그림 6은 태그간격의 분포를 고려하지 않고 LID를 할당했을 때 질의 성능저하의 문제를 보여주고 있다. 그림 6(a)와 같이 TID_i가 LID_{20}, LID_{40}, LID_{60}의 순서로 위치를 이동했을 때 태그간격 $TI_{i,1}$, $TI_{i,2}$, $TI_{i,3}$이 생성되었다고 가정하자. 그림 6(b)는 이와 같이 생성된 태그간격을 R-tree에 저장했을 때 노드 구조에 대한 예를 보여주고 있다. 생성된 태그간격은 도메인 공간에서 서로 멀리 떨어져 있기 때문에 각 태그간격을 저장하고 있는 단말 노드 R1, R2, R3 역시 서로 멀리 떨어져 있을 가능성이 크다. 이때, 궤적 질의 TQ = (*, TID_i, $[t_1^{\vdash}, t_4^{\dashv}]$)가 수행된다면, $TI_{i,1}$, $TI_{i,2}$, $TI_{i,3}$를 찾기 위한 탐색 경로(traversing path)가 길어져 높은 노드 접근 비용이 필요하게 될 것이다.

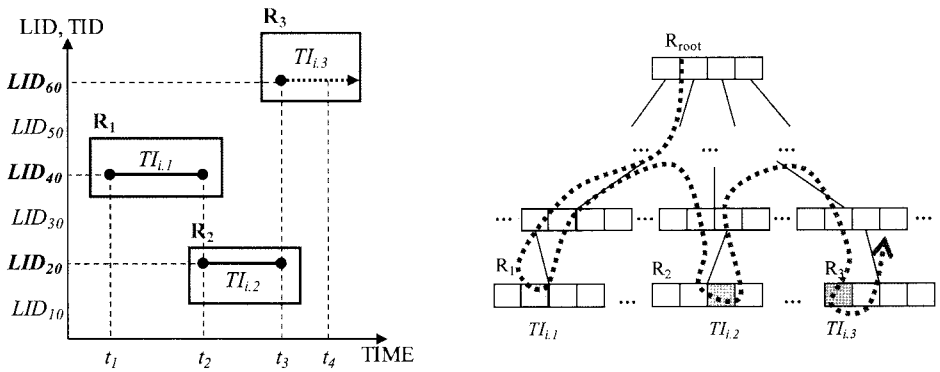
이러한 문제를 해결하기 위하여 이 논문에서 제시하는 기본 아이디어는 인식공간에 부여된 LID를 태그 위



(a) LID 순서가 태그 이동경로를 따르지 않을 경우 설정된 검색범위

(b) LID 순서가 태그 이동경로를 따를 경우 생성된 검색범위

그림 5 LID 도메인에서의 LID 순서에 따른 검색범위의 차이



(a) 3차원 공간상에서 LID의 배치에 따른 태그간격 정보의 표현

(b) 서로 멀리 떨어진 R-tree의 단말노드에 저장된 태그간격 정보의 예제

그림 6 태그간격의 분포를 고려하지 않은 LID 할당의 문제점

치추적 질의 시 검색영역을 줄일 수 있도록 LID 도메인에 재정렬하는 것이다. 이는 두 LID가 도메인에서 얼마나 가까이 정렬될 수 있는지를 결정하는 LID 근접성 함수를 기반으로 수행된다.

4. 위치 식별자 근접성

LID 간의 근접성을 정의하기 위해서 이 장에서는 먼저 LID 근접성 결정에 주요 요소로 작용하는 흐름경로에 대해서 살펴본다. 또한, 이를 기반으로 생성된 흐름경로 그래프를 이용하여 LID간의 근접성을 정의하고 이를 계산하기 위한 근접성 함수를 제시한다.

4.1 흐름경로

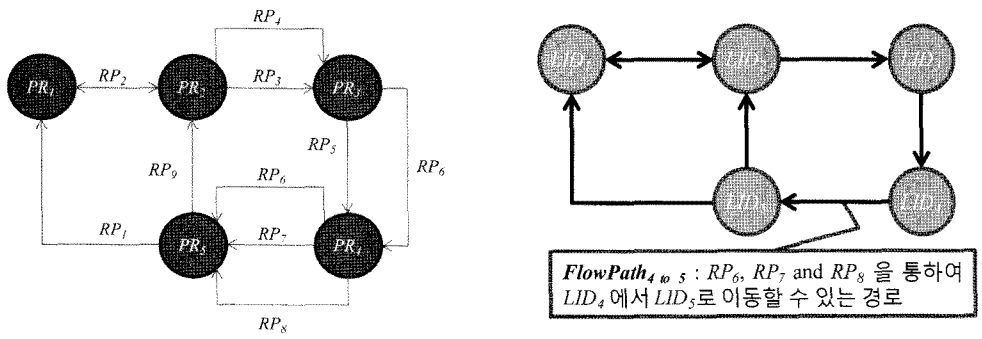
태그를 부착한 물품은 인식공간의 입출력을 담당하는 판독점을 통과함으로써 다른 인식공간으로 이동을 한다. 그러나, 물품들은 모든 인식공간 사이를 자유롭게 이동할 수 있는 것은 아니다. 인식공간 사이의 이동은 RFID 시스템의 비즈니스 흐름 또는 인식공간 간의 물리적인 환경에 의해서 제한될 수 있다. 예를 들어, 공급망 관리 시스템에서는 특정 공장에서 생산된 제품은 미리 등록된 물류 센터와 도매상, 소매상 간에 이동이 이루어지며 다른 지점으로의 이동은 제한된다. 또한, 물류 센터의 저장 창고에 있는 물품이 도매상 또는 소매상으로 출고되기 위해서는 항상 출고 창고로 옮겨져야 한다. 이와 같이, RFID 시스템에서 태그를 부착한 물품은 모든 인식공간 사이에 미리 정해진 흐름경로(path of tag flows: FlowPath)를 따라 이동을 하며 이를 통하여 태그의 흐름이 생성된다. 즉, 태그가 LID_i에서 LID_j로 이동을 하고자 한다면 흐름경로 $FlowPath_{i\ to\ j}$를 통해서 이동해야 한다.

흐름경로는 두 인식공간 사이의 연결 정보를 표현하기 위한 간략한 표현방법으로 그림 7과 같이 인식공간과 판독점으로 이루어진 연결 그래프를 통해서 생성이 가능하다. 그림 7(a)의 인식공간 PR_1에서 PR_5는 흐름경로 표현에서 LID_1에서 LID_5로 각각 대응된다. 또한, 각 인식공간을 연결하는 하나 이상의 판독점은 흐름경로 표현에서 단일 연결선으로 표현된다. 예를 들어, 그림 7(a)의 PR_4와 PR_5 사이에는 PR_4에서 PR_5로 이동할 수 있는 RP_6, RP_7, RP_8의 판독점으로 연결되어 있으며, 이는 그림 7(b)와 같이 LID_4에서 LID_5 방향으로 연결되는 단일 연결선으로 표현된다.

흐름경로는 다음과 같은 속성을 가진다.

1) 흐름경로는 방향성을 가진 경로(directional path)이다. 판독점은 태그의 입력, 출력, 입출력의 세 가지 방향성을 제공하며 흐름경로는 각 판독점의 방향성과 동일한 방향을 가진다.
2) 모든 LID는 최소 하나 이상의 흐름경로와 연결되어 있다. 즉, 인식공간은 태그가 이동할 수 있는 모든 RFID 공간을 표현하므로 임의의 인식공간에서 다른 인식공간으로 이동할 수 있는 판독점은 항상 존재한다.
3) 임의의 두 LID를 직접 연결하는 흐름경로가 없을 수도 있다. 이러한 경우, 태그는 두 LID 사이를 이동하기 위해서 흐름경로로 연결된 하나 이상의 다른 LID를 통과해야 한다.

태그는 각 인식공간 사이를 흐름경로를 따라 이동하므로 흐름경로는 태그의 이동 특성을 결정하는 기본 요소이며, 흐름경로를 통한 이동 패턴에 따라서 데이터 공간에서 태그간격의 분포에도 영향을 미치게 된다. 3장에서 살펴보았듯이 태그간격의 분포는 도메인에서 LID의 정렬에 따라 변할 수 있으며, 이에 따라 태그 위치추적 질의의 성능이 달라진다. 즉, 동일 흐름경로를 따라서 위치하는 인식공간 사이를 이동하는 태그를 통하여 생



(a) 인식공간과 인식공간을 연결하는 판독점 간의 연결 그래프 예제 (b) 연결 그래프를 이용한 흐름경로의 생성

그림 7 연결 그래프를 이용한 흐름경로의 생성

성된 태그간격이 다른 태그간격보다 함께 접근될 확률이 높다. 따라서, 흐름경로와 그에 따라 발생한 태그간격의 분포는 질의의 성능과 밀접한 관련이 있다.

4.2 위치 식별자 근접성의 정의

색인의 도메인에 LID를 재정렬하기 위하여 먼저 LID 간의 근접성을 다음과 같이 정의한다.

정의 1. LID 근접성 (*LID Proximity: LIDProx*)

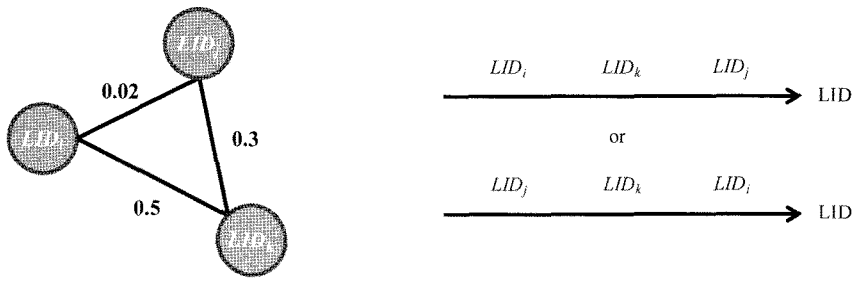
태그간격 색인의 LID 도메인에서 임의의 두 LID 간의 근접도 값(closeness value)을 **LID 근접성**이라 정의한다. 임의의 두 LID_i, LID_j의 LID 근접성은 $LIDProx_{ij}$ 또는 $LIDProx_{ji}$로 표현한다.

LID 근접성은 다음과 같은 속성을 가지고 있다.

1) LID 도메인에서 임의의 LID_i는 모든 LID_j에 대해서 $LIDProx_{ij}$와 $LIDProx_{ji}$를 가진다. (단, $i \neq j$)
2) LID 도메인을 구성하는 모든 LID_i, LID_j에 대해서 $LIDProx_{ij}$와 $LIDProx_{ji}$의 값은 동일하다.
3) 만약 $LIDProx_{ij} < LIDProx_{ik}$인 LID_k가 존재하지 않는다면, LID_i와 가장 근접성 LID는 LID_j이다.

그림 8은 LID 간에 부여된 LID 근접성에 의해 LID 도메인에 정렬되는 예제를 보여주고 있다. 그림 8(a)와 같이 LID_i, LID_j, LID_k가 존재하고 $LIDProx_{ij}$, $LIDProx_{jk}$, $LIDProx_{ki}$가 각각 0.02. 0.3. 0.5라고 한다면, 세 LID에 대해서 가장 LID 근접성이 높은 정렬 순서는 (LID_i, LID_k, LID_j) 또는 (LID_j, LID_k, LID_i)가 되며 그림 8(b)와 같이 정렬될 수 있다. LID 근접성의 정의와 속성에 따라 그림 8(b)와 같이 정렬된다면 색인에 저장된 태그간격의 동시 접근 확률이 최대가 되며 이에 따라 효율적인 질의 처리가 가능하다.

LID 근접성은 흐름경로를 기반으로 한 그래프 구조를 이용하여 표현하는 것이 가능하며 그래프는 다음과 같은 속성을 가져야 한다. 첫째, 그래프는 가중치(weighted) 그래프가 되어야 한다. 그래프의 정점을 구성하는 모든 LID에 대해서 LID 근접성을 표현하기 위해서는 각 정점 간의 가중치가 표현되어야 하며, 이를 위해서 각 정점을 연결하는 연결선에 가중치가 부여되어야 한다. 둘째, 그래프는 완전(complete), 무방향(undirected) 그래프이어야 한다. 흐름경로를 구성하는 LID들은 LID 근접성의 속성 1)에 따라 그래프의 정점을 구성하는 모든 LID들은 연결선을 통하여 LID 근접성이 부여되어야 한다. 또한, 속성 2)에 의해서 LID 근접성은 오직 대상에 되는 두 LID에서



(a) LID_i, LID_j, LID_k 간에 부여된 LID 근접성 (b) LID 근접성에 따른 도메인 상의 최적 LID 정렬

그림 8 LID 근접성을 이용한 LID 도메인 정렬 예

생성된 태그간격의 동시 접근 확률을 표현하기 때문에 두 LID의 접근 순서는 고려할 필요가 없다.

4.3 위치 식별자 근접성 함수

이 절에서는 LID 근접성을 계산하기 위하여 사용될 LID 근접성 함수를 제시한다. 임의의 LID_i와 LID_j 간의 LID 근접성을 구하기 위한 LID 근접성 함수 LID Prox(i, j)는 흐름경로와 정적 속성을 기반으로 한 정적 LID 근접성(Static LID Proximity: SLIDProx) 함수와 시간이 흐름에 따라 계속해서 변경되는 동적 속성을 기반으로 한 동적 LID 근접성(Dynamic LID Proximity: DLIDProx) 함수로 나누어진다.

4.3.1 정적 위치 식별자 근접성 함수

정적 LID 근접성 함수는 흐름경로에서 두 인식공간을 연결하는 판독점의 입출력 방향 개수 - 흐름경로 수(FlowPath Cardinality: FC) - 를 통하여 계산된다. 태그는 임의의 두 인식공간 사이를 이동하기 위해 판독점을 통과하기 때문에 태그가 출입할 수 있는 판독점이 많으면 판독점이 적은 인식공간보다 단위 시간 당 보다 많은 수의 태그가 이동할 수 있으며 결과적으로 많은 태그간격의 생성이 가능하다. 따라서, 임의의 두 LID 사이의 흐름경로 수는 해당 LID에 대응되는 두 인식공간에서의 태그간격 생성량과 밀접한 관련이 있다.

임의의 LID_i에서 LID_j로 연결된 흐름경로 수를 $FC_{i\ to\ j}$라고 할 때 LID_i와 LID_j 사이에서 존재 가능한 흐름경로 수는 $FC_{i\ to\ j}$와 $FC_{j\ to\ i}$의 합으로 구할 수 있다. 모든 인식공간은 하나 이상의 판독점을 가지고 있지만 두 인식공간 사이를 연결하는 판독점이 없을 경우 대응되는 LID 간의 흐름경로 수는 0이 되며, $FC_{i\ to\ j}$, $FC_{j\ to\ i}$가 각각 임의의 개수 M, N이라고 할 때 LID_i와 LID_j 사이의 흐름경로 수는 (M + N)이 된다.

LID의 집합 LIDSet = {LID_1, LID_2, …, LID_n}에 n개의 LID가 존재한다고 했을 때 흐름경로 수 FC를 기반으로 임의의 LID_i와 LID_j 간의 정적 LID 근접성 함수 SLIDProx(i, j)를 식 (1)과 같이 정의한다(단, $i \neq j$이며 LIDSet에 존재하는 임의의 LID_i에 대하여 $FC_{i\ to\ i}$ = 0).

$$\mathrm{SLIDProx}(i,j)=\left(FC_{i\ to\ j}+FC_{j\ to\ i}\right)\Big/\sum_{a=1}^{n}\left(\sum_{b=1}^{n}FC_{a\ to\ b}\right) \quad (1)$$

식 (1)을 통해서 SLIDProx(i, j)에 의해서 계산되는 $LIDProx_{ij}$는 대상 RFID 시스템의 모든 흐름경로에 대해서 LID_i와 LID_j 사이의 흐름경로의 비율로 계산됨을 알 수 있다. 즉, 임의의 두 LID를 선택했을 때 이들간의 흐름경로 수가 많을수록 LID 근접성은 커진다. 반대로, 연결하는 흐름경로가 없을 경우에는 LID 근접성은 0이 된다.

4.3.2 동적 위치 식별자 근접성 함수

흐름경로를 통한 태그의 이동량과 이를 추적하는 태그 위치추적 질의의 종류 및 빈도는 시간의 흐름에 따라 계속해서 변경된다. 이는 데이터 공간에 분포하는 태그간격에 대한 동시 접근 확률 역시 시간이 흐름에 따라 계속해서 변경됨을 의미한다. 앞서 살펴본 정적 LID 근접성 함수는 두 LID를 연결하는 판독점의 개수 또는 속성이 변경되지 않는다면 근접성은 변경되지 않는다. 따라서, 정적 LID 근접성 함수는 동적으로 변경되는 LID 근접성의 변화에 대처하지 못한다.

이 절에서는 시간이 흐름에 따라 동적으로 값이 변경되는 LID 근접성을 계산하기 위하여 동적 LID 근접성 함수를 시간 매개 함수(time parameterized function) 형태로 제시한다. 식 (2)는 임의의 LID_i와 LID_j에 대해서 임의의 T 시간에 LID 근접성을 계산하기 위한 동적 LID 근접성 함수 $\mathrm{DLIDProx}_T(i, j)$를 정의하고 있다(단, $i \neq j$). $\mathrm{LIDProx_OQ}_T(i, j)$와 $\mathrm{LIDProx_TQ}_T(i, j)$는 각각 관측 질의와 궤적 질의에 대한 LID 근접성을 계산하기 위한 함수이며, a는 두 근접성 함수에 대한 가중치이다.

$$\mathrm{DLIDProx}_T(i,j)=\alpha\times\mathrm{LIDProx\_OQ}_T(i,j)+(1-\alpha)\times\mathrm{LIDProx\_TQ}_T(i,j) \quad (2)$$

임의의 두 LID_i와 LID_j에 대한 LID 근접성은 수행되는 태그 위치추적 질의의 종류에 따라 달라진다. 따라서, $\mathrm{DLIDProx}_T(i, j)$는 $\mathrm{LIDProx_OQ}_T(i, j)$와 $\mathrm{LIDProx_TQ}_T(i, j)$를 통하여 질의에 따른 LID 근접성을 모두 반영한다. 그리고, a를 통하여 두 함수에 대한 가중치를 부여함으로써 $\mathrm{LIDProx_OQ}_T(i, j)$와 $\mathrm{LIDProx_TQ}_T(i, j)$의 반영비율을 결정한다. 두 질의에 대한 반영 비율은 식 (3)에서와 같이 T 시간까지 LID_i와 LID_j를 접근하는 전체 질의 중 해당 질의의 비율로써 결정된다.

$$\alpha=\begin{cases}0 \text{ or } 1 & \text{if no queries are processed for } LID_i \text{ and } LID_j\\ \sum_{t=1}^{T}OQ_{ij,t}\Big/\sum_{t=1}^{T}(OQ_{ij,t}+TQ_{ij,t}) & \text{otherwise}\end{cases} \quad (3)$$

관측 질의에 대한 LID 근접성은 관측 질의 시 함께 접근되는 두 LID에서 생성된 태그간격의 개수에 영향을 받는다. LID_i, LID_j, LID_k에서 t 시간에 생성된 태그간격을 각각 $TI_{i,t}$, $TI_{j,t}$, $TI_{k,t}$라고 하며, 각 LID에서의 태그간격 생성개수가 $\sum_{t=1}^{T}TI_{i,t}>\sum_{t=1}^{T}TI_{j,t}>\sum_{t=1}^{T}TI_{k,t}$ 의 비율이라고 가정하자. 이때, 각 LID에 대해서 동일한 비율의 관측 질의 수행 시, 정의 1에 의해서 LID_i와 LID_j에서 생성된 태그간격의 동시접근 확률이 가장 높아야 한다. 이에 따라, 관측 질의에 대한 LID 근접성 함수 $\mathrm{LIDProx_OQ}_T(i, j)$는 태그간격 생성 개수에 따른 근접성의 변화를 반영하기 위해서 식 (4)와 같이 전체 태그

간격 생성 개수 대비 LID_i, LID_j에서 생성된 태그간격 개수의 비율로써 계산된다(단, σ_{OQ} 및 δ_{OQ}는 태그간격 개수의 비율에 대한 가중치).

$$\text{LIDProx\_OQ}_T(i,j)=\frac{\delta_{OQ}}{\sigma_{OQ}}\times\left\{\sum_{t=1}^{T}\left(TI_{i,t}+TI_{j,t}\right)\Big/\sum_{t=1}^{T}\sum_{a=1}^{n}TI_{a,t}\right\} \quad (4)$$

관측 질의에 대한 LID 근접성은 두 LID에서 동일한 개수의 태그간격이 생성되었어도 시간 도메인에서의 태그간격의 분포에 의해서 달라질 수 있다. 즉, 시간대별 분포의 편차에 의해서 관측 질의가 수행되었을 때 대상 태그간격의 접근 확률이 달라지게 된다.

그림 9와 같이 시간 t_1부터 시간 t_n까지 LID_i와 LID_j, LID_i와 LID_k, LID_i와 LID_l에서 생성된 태그간격의 개수가 각각 (a), (b), (c)의 분포를 보이며, 태그간격 분포 (a), (b), (c)에서 생성된 태그간격의 시간대별 평균 개수는 M으로 동일하다고 가정하자. 이때, 각 태그간격 분포에 대해서 동일한 시간 조건자를 가지는 임의의 관측 질의 OQ_{ij}, OQ_{ik}, OQ_{il}이 수행된다면 시간 조건자의 범위에 따라서 함께 접근되는 태그간격의 개수가 달라진다. 즉, 시간 조건자가 t_i 부근에 위치하고 있다면 (c)의 분포에서 태그간격의 동시 접근확률이 가장 높아지며, LID_i와 근접성이 가장 높은 LID는 LID_l이 된다. 그러나, 시간 조건자가 t_i에서 멀리 떨어져 있다면 OQ_{il} 수행 시 태그간격 동시 접근확률이 OQ_{ij}, OQ_{ik}보다 떨어지며, 결과적으로 LID_l은 LID_j, LID_k 보다 LID_i와의 근접성이 낮아진다. 따라서, 모든 시간 범위에서 일정한 태그간격의 동시 접근확률을 가지려면 태그간격 분포 (a)와 같이 고른 태그간격의 분포를 가져야 한다.

시간 도메인에서 태그간격 분포의 차이는 태그간격 생성수의 표준편차로 표현된다. 식 (5)의 σ_{OQ}는 $\overline{TI}_i$, $\overline{TI}_j$가 각각 LID_i, LID_j에서 생성된 태그간격의 평균 개수라고 할 때 T 시간까지 LID_i와 LID_j에 의해서 생성된 태그간격의 표준편차를 나타내고 있다. 태그간격 분포는 표준편차가 작을수록 그림 9의 (a)와 같이 고른 분포를 나타내므로 σ_{OQ}는 관측 질의에 대한 LID 근접성과 반비례하며, 식 (4)에서 태그간격 생성개수의 비율에 반비례하는 가중치로 사용된다.

$$\sigma_{OQ}=\sqrt{\frac{1}{T}\times\sum_{t=1}^{T}\left\{\left(TI_{i,t}+TI_{j,t}\right)-\left(\overline{TI}_i+\overline{TI}_j\right)\right\}^2} \quad (5)$$

마지막으로, 관측 질의에 대한 LID 근접성은 질의결과에 포함되는 태그간격의 비율을 동시에 고려해야 한다. 예를 들어, 그림 9의 (c)가 모든 시간 범위를 대상으로 했을 때 (a)와 (b)에 비해서 태그간격의 접근 확률이 떨어진다고 하더라도, 대부분의 관측 질의 OQ_{il}의 시간 조건자가 t_i 주위에 집중되어 있다면 (a), (b) 보다 많은 수의 태그간격을 결과로 포함하게 된다. 따라서, 관측 질의에 대한 LID 근접성은 질의결과에 포함되는 태그간격의 비율에 비례하여 증가하여야 한다. 식 (6)의 δ_{OQ}는 T 시간까지 관측 질의 OQ_{ij}가 수행되었을 때 LID_i와 LID_j에서 생성된 태그간격이 질의결과에 포함되는 비율을 나타내고 있으며, 식 (4)에서 태그간격 생성개수의 비율에 비례하는 가중치로 사용된다(단, STI_i, STI_j는 각각 LID_i, LID_j에서 생성된 태그간격 중 질의결과에 포함된 태그간격의 개수의 총합).

$$\delta_{OQ}=\left\{\left(STI_i+STI_j\right)\Big/\sum_{t=1}^{T}\left(TI_{i,t}+TI_{j,t}\right)\right\}\times\left(1\Big/\sum_{t=1}^{T}OQ_{ij,t}\right) \quad (6)$$

궤적 질의에 대한 LID 근접성은 관측 질의와는 달리 LID 사이의 태그 이동량의 영향을 받는다. 즉, LID_i 또는 LID_j에 머무르는 태그를 추적하는 궤적 질의가 수행되었을 때, LID_i 또는 LID_j에 머물렀던 대부분의 태그가 각각 LID_j 또는 LID_i로 이동한다면 해당 태그에 대한 궤적 질의 시 LID_i와 LID_j가 함께 접근될 확률은 높아진다. 식 (7)은 임의의 LID_i와 LID_j에 대해서 T 시간에서의 궤적 질의에 대한 LID 근접성 함수 LIDProx_TQ$_T(i, j)$를 제시하고 있다. LIDProx_TQ$_T(i, j)$는 생성된 태그간격 중 두 LID 사이를 이동한 태그간격의 비율로써 계산된다(단, $TM_{i\ to\ j,t}$는 t 시간에 LID_i에서 LID_j로 이동한 태그의 이동량).

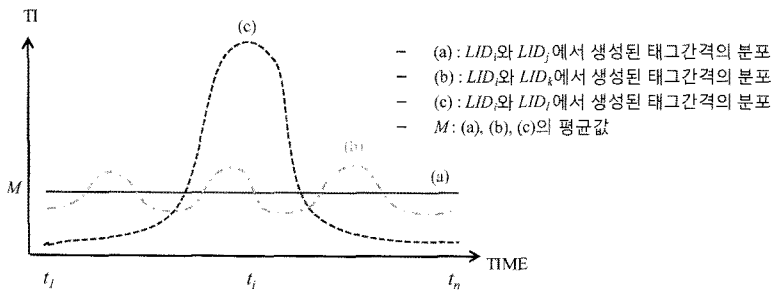


그림 9 시간의 흐름에 따른 태그간격의 분포 예제

$$\text{LIDProx\_TQ}_T(i,j) = \frac{\delta_{TQ}}{\sigma_{TQ}} \times \frac{\sum_{t=1}^{T}\left(TM_{i\,to\,j,t} + TM_{j\,to\,i,t}\right)}{\sum_{t=1}^{T}\left(\sum_{a=1}^{n}\sum_{b=1}^{n} TM_{a\,to\,b,t} - \sum_{c=1}^{n} TM_{c\,to\,c,t}\right)} \quad (7)$$

관측 질의에서와 마찬가지로 궤적 질의에 대한 LID 근접성은 시간 도메인에서의 태그간격의 분포 및 질의결과에 포함된 태그간격의 비율에 의해서도 영향을 받는다. 다만, 궤적 질의는 LID에서 생성된 태그간격의 개수가 아닌 두 LID 사이의 태그 이동량에 영향을 받기 때문에 두 LID 사이에서의 태그 이동량의 분포가 적용되어야 한다. 이를 반영하기 위하여, 식 (8)과 식 (9)에서는 시간 도메인 상에서 LID_i와 LID_j 간 태그 이동량의 표준편차를 계산하는 σ_{TQ}와 궤적 질의 시 LID_i와 LID_j를 포함하는 태그간격의 비율을 계산하는 δ_{TQ}를 나타내고 있다. 관측 질의에서와 마찬가지로 전체 시간 도메인에서 고른 태그 이동량의 분포를 보이며 태그간격이 질의 결과에 포함되는 비율이 높을수록 궤적 질의에 대한 두 LID의 접근 확률이 높아지기 때문에 식 (7)에서 σ_{TQ}는 태그 이동량 비율에 반비례하는 가중치로, δ_{TQ}는 태그 이동량 비율에 비례하는 가중치로 각각 사용된다.

$$\sigma_{TQ} = \sqrt{\frac{1}{T}\times\sum_{t=1}^{T}\left\{\left(TM_{i\,to\,j,t} + TM_{j\,to\,i,t}\right) - \left(\overline{TM}_{i\,to\,j} + \overline{TM}_{j\,to\,i}\right)\right\}^2} \quad (8)$$

$$\delta_{TQ} = \left\{\sum_{t=1}^{T}\left(STI_{i,t} + STI_{j,t}\right) \Big/ \sum_{t=1}^{T}\left(TI_{i,t} + TI_{j,t}\right)\right\} \times \left(1 \Big/ \sum_{t=1}^{T} TQ_{ij,t}\right) \quad (9)$$

5. 위치 식별자 재순서 기법

이 장에서는 4장에서 제시한 LID 근접성 함수를 기반으로 LID 재순서화 문제를 정의하고, 이를 해결하기 위한 방법을 제시한다.

LID의 개수가 n인 LID 집합 LIDSet = {LID_1, LID_2, …, LID_{n-1}, LID_n}을 색인의 도메인에 정렬하기 위해서 임의의 LID 순서 리스트 $OLIDList_i$ = ($OLID_{i,1}$, $OLID_{i,2}$, …, $OLID_{i,n-1}$, $OLID_{i,n}$)을 생성할 수 있다. LID 순서 리스트는 LIDSet으로부터 $OLIDList_1$부터 $OLIDList_{n!/2}$까지 $n!/2$개의 조합을 추출하는 것이 가능하다. 태그 위치추적 질의 수행 시 태그간격의 동시 접근 확률이 최대가 되는 LID 순서 리스트를 찾기 위해서 먼저 다음과 같이 LID 순서 리스트에 대한 선형 근접성을 정의한다.

정의 2. 선형 근접성(*Linear Proximity: LinearProx*)

$OLIDList_a$를 구성하는 모든 OLID에 대하여 인접한 OLID 간 LID 근접성의 합을 $OLIDList_a$의 **선형 근접성(LinearProxa)**이라 정의하며 다음의 수식으로 계산한다.

$$LinearProx_a = \sum_{i=1}^{n-1} \text{LIDProx}(a.i, a.i+1) \quad (10)$$

그림 10과 같이 LID_1, LID_2, LID_3, LID_4를 포함하는 LID 집합이 있다고 가정하자. LID 순서 리스트는 $OLIDList_1$부터 $OLIDList_{12}$까지 12개(4!/2)가 생성될 수 있으며, 정의 2에 의해서 모든 LID 순서 리스트는 선형 근접성을 가진다. 예를 들어, $OLIDList_9$는 $LinearProx_9$ = LIDProx(2, 3) + LIDProx(3,1) + LIDProx(1, 4)의 선형 근접성을 가진다.

질의 시 함께 접근되는 태그간격의 동시 접근확률이 최대가 되기 위해서는 도메인 상에서 근접해서 정렬되는 모든 LID들은 항상 가장 큰 LID 근접성을 가져야 한다. 이는 태그 위치추적 질의가 i개의 LID에서 생성된 태그간격을 동시 접근한다고 가정하면, 대응되는 LID가 모두 인접해서 정렬되며 인접한 LID 간의 LID 근접성의 합이 최대이어야 함을 의미한다. 따라서, 임의의 LID들을 접근하는 모든 태그 위치추적 질의에 대해서 최대의 태그간격 동시 접근확률을 얻기 위해서는 선형 근접성이 최대가 되는 LID 순서 리스트를 생성해야 한다. 이를 해결하기 위하여 다음과 같이 LID 재순서화 문제를 정의한다.

정의 3. LID 재순서화 문제(*LID reOrdering Problem: LOP*)

n개의 LID를 포함하는 LID 집합 LIDSet = {LID_1, LID_2, …, LID_{n-1}, LID_n}과 각 LID 간의 LID 근접성을 이용하여, 선형 근접성이 최대가 되는 LID 순서 리스트 $OLIDList_o$ = ($OLID_{o,1}$, $OLID_{o,2}$, …, $OLID_{o,n-1}$, $OLID_{o,n}$)을 결정하는 것을 **LID 재순서화 문제**라고 정의한다.

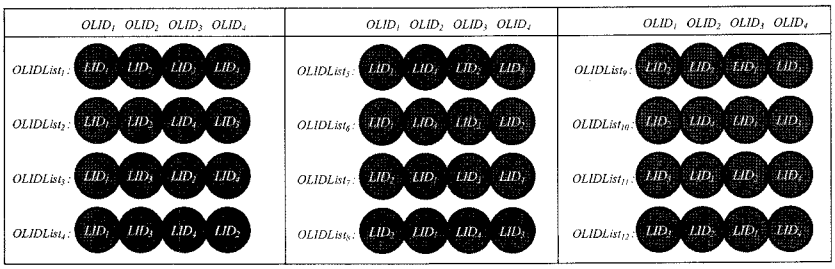


그림 10 LID 순서 리스트의 조합 예제

LOP는 시작점과 종료점을 가지지 않는 최소 가중치 해밀턴 경로 문제(Minimal Weighted Hamiltonian Path Problem: MWHP)와 매우 유사하므로 가중치 그래프를 이용하여 MWHP를 해결하기 위한 기존의 방법을 적용할 수 있다. 그러나, MWHP는 가중치 그래프에서 최소 가중치를 가지는 해밀턴 사이클(Hamiltonian Cycle: HC)을 찾지만, 정의 2에 의해서 LOP는 최대 가중치를 가지는 사이클을 찾아야 한다. 따라서, LOP를 MWHP에 적용하기 위해서는 가중치 그래프에서 LID_i와 LID_j 간의 가중치를 1 - LIDProx(i, j), 또는 1 - LIDProx(j, i)로 변경하는 것이 필요하다. LOP는 정리 1에 의해서 외판원 여행 문제(Traveling Salesman Problem: TSP)를 이용하여 해결할 수 있다.

정리 1. LOP는 다음과 같이 대응되는 가중치 그래프 $G = (V, E, w)$에 대해서 TSP와 동일하다.

- 정점 V = LIDSet $\cup$ {v_0} (단, 임의의 정점 v_0는 MWHP를 TSP로 풀기 위해 추가된 정점)
- 연결선 E = {(LID_i, LID_j) | LID_i, LID_j $\in$ LIDSet, $i \neq j$} $\cup$ {(LID_i, v_0) | LID_i $\in$ LIDSet)
- 가중치 w: $E \rightarrow R$, $w(i, j)$ = 1 - LIDProx(i, j) = 1 - LIDProx(j, i) = $w(j, i)$, $w(i, v_0)$ = $w(v_0, i)$ = 0

증명

그래프 G는 완전 가중치 그래프이기 때문에 해밀턴 사이클 HC를 포함하고 있다. 그래프 G에 의해서 생성된 최소 가중치 해밀턴 사이클을 HC_{min}이라고 가정하자(단, HC_{min} = {(v_0, $OLID_{a1}$), ($OLID_{a1}$, $OLID_{a2}$), ..., ($OLID_{an-1}$, $OLID_{an}$), ($OLID_{an}$, v_0)}이며 $OLID_{ai}$ $\in$ LIDSet). 만약 정점 v_0를 포함하고 있는 두 링크 (v_0, $OLID_{a1}$)과 ($OLID_{an}$, v_0)가 HC_{min}으로부터 제거된다면, 그래프 G에서 $OLID_{a1}$로부터 $OLID_{an}$까지 최소 가중치로 방문할 수 있는 최소 가중치 해밀턴 경로 L_{min}을 얻을 수 있다. 경로 L_{min}을 만들기 위해서 제거된 모든 링크는 정점 v_0를 포함하고 있으며 이 링크들의 가중치는 0이기 때문에 사이클 HC_{min}의 가중치는 경로 L_{min}의 가중치와 동일하다. 또한, 생성된 경로 L_{min}은 순서화된 OLID들의 리스트인 $OLIDList_a$로 변경될 수 있다 (단, $OLIDList_a$ = ($OLID_{a1}$, $OLID_{a2}$, ..., $OLID_{an-1}$, $OLID_{an}$)). 이와 같은 이유에 의해서, LOP는 그래프 G에서 HC_{min}을 구하기 위해 대응되는 TSP와 동일하다고 할 수 있다.

TSP는 가장 유명한 비결정 완전(NP-complete) 문제 중의 하나이다[21]. 만약 n개의 정점을 가진 그래프 G에 대해서 최소의 비용을 가진 경로를 찾고자 한다면, 그래프 G에서 생성될 수 있는 $n!$개의 해밀턴 사이클을 모두 조사해야 한다. 그러나, n개의 정점을 방문하기 위해서 모든 가능한 경우를 조사하는 것은 철저한 탐색(exhaustive exploration)을 필요로 하므로 상당히 비효율적이다. 예를 들어, 50개의 정점을 한번씩 방문하는 경로를 찾기 위해서는 3.04×10^{64}개의 서로 다른 해밀턴 사이클을 조사해야 하고, 이는 엄청난 양의 계산 시간을 필요로 한다.

TSP 문제를 풀기 위해서 이전 연구에서는 유전 알고리즘(Genetic Algorithm: GA), 담금질 알고리즘(Simulated Annealing: SA), 신경망 알고리즘(Neural Network: NN)과 같은 여러 가지의 휴리스틱 기반 기법들을 제시하였다. 휴리스틱 기반 기법들은 적은 시간으로 비결정 완전 문제를 풀기 위한 해를 찾는데 사용될 수 있다. 이러한 기법들은 최적의 해(best solution)를 구하지는 못하지만 최적의 해에 가까운 지역 최적해(local optima)를 찾을 수 있기 때문에 많은 응용에서 비결정 완전 문제를 풀기 위해서 사용된다[21].

6. 실험 평가

이 장에서는 논문에서 제안하는 LID 근접성을 기반으로 한 LID 재순서 기법의 성능을 평가한다. 성능 평가를 위하여 태그간격 색인에서 태그 위치추적 질의를 수행한다. 색인의 도메인으로 사용하기 위하여 기존의 방식으로 인식공간에 LID를 부여한 색인을 구축한 후 LID 근접성을 이용하여 LID를 재정렬하고 각 색인들에 대한 질의의 성능을 비교한다. 질의 성능을 비교하기 위한 색인으로는 태그간격의 저장 및 검색을 지원하기 위해 제시된 TPIR-tree[8]와 이동체의 저장 및 검색을 위한 R*-tree[22], TB-tree[10]를 사용하였다. 각 색인은 LID, TID, TIME을 도메인으로 하는 3차원 공간에서의 태그간격 데이터 모델을 기반으로 구축하였다. 또한, 각 색인의 기본 속성을 그대로 유지하기 위하여 해당 색인의 삽입 및 분할 정책을 그대로 사용하였다. LID 순서 리스트를 생성하기 위하여 이 논문에서는 여러 가지 휴리스틱 기반 기법들 중에서 GA[23]를 사용한다. GA는 TSP 문제를 포함하여 조합 최적화(combinatorial optimization) 문제를 풀기 위해서 실제 응용에서 많이 적용되고 있다[24,25].

6.1 실험 설정

이 논문에서 제안하는 LID 근접성을 평가하기 위하여 사용된 색인은 Java 2 Platform Standard Edition 5.0을 이용하여 구현하였다. 또한, LID 근접성을 기반으로 LID를 재정렬하기 위하여 사용된 TSP 솔루션인 GA는 Microsoft Visual C++ 2005를 사용하여 구현하였다. 실험은 Windows XP Professional 운영체제 기반의 2GB 메인 메모리, CPU Pentium IV 2.6GHz를 장착한 PC를 이용하여 수행하였다.

RFID 시스템에서는 기존의 이동체 데이터베이스에서의 GSTD 데이터 셋[26] 또는 네트워크 데이터 생성기[27]와 같이 색인 성능 측정을 위해 널리 사용되고 있는 데이터 셋이 존재하지 않는다. 이에 본 논문에서는 성능 평가를 위한 태그 데이터 생성기(Tag Data Generator: TDG)를 제작하였으며, TDG를 통하여 임의의 데이터

셋(synthetic dataset)을 생성하였다. TDG는 색인에 저장할 태그간격을 생성하기 위하여 인식공간과 및 판독점의 개수를 임의로 지정할 수 있으며, 인식공간 간의 판독기의 연결 개수 및 방향, 태그의 이동 구간, 초기 태그 생성 개수, 총 태그 생성 개수, 시간 당 태그 이동 비율 등을 설정할 수 있다. 또한, 실제 RFID 환경을 반영하기 위해서 태그의 생성(공장, 농산물 생산지 등)과 태그의 소멸(소매점, 상품 폐기장 등) 역할을 할 수 있는 인식공간을 임의로 지정할 수 있다. 이에 따라, 태그는 인식공간 사이를 이동하면서 반복적으로 생성되고 소멸된다.

표 2 TDG에 의해서 생성된 데이터 셋의 종류

종류	초기 태그 생성 개수	총 태그 생성 개수	생성된 태그간격 개수
100K	200	1,000	100,462
200K	200	1,000	200,297
300K	200	1,000	300,274
400K	200	1,000	400,921
500K	200	1,000	500,378

실험을 위하여 인식공간의 개수가 200개이고 각 인식공간 사이의 판독점의 개수를 0에서 5까지 랜덤으로 부여한 RFID 공간(*PR200*)을 사용하였다. 표 2는 *PR200*을 이용하여 생성된 태그간격 데이터 셋이다. 이 실험에서 평가하고자 하는 것은 LID 재순서화에 따른 색인의 성능이므로 적용된 LID의 순서에 따라서 LID 도메인의 정렬이 변경된다. 따라서, 데이터 공간에서 태그간격을 특정 분포 - 가우시안 분포(Gaussian distribution), 사향 분포(skewed distribution) 등 - 를 모든 색인에 대해서 동일하게 생성하는 것이 불가능하다. 표 2의 데이터 셋에 대해서 데이터 공간에서 태그간격이 균등 분포를 이루도록 하기 위해 초기 200개의 태그가 태그 생성을 담당하는 인식공간에 랜덤하게 분포되고 인식공간의 연결속성에 따라서 랜덤하게 이동하도록 하였다. 시간이 지남에 따라 태그간격이 데이터 공간에서 랜덤하게 생성되도록 하여 데이터 셋이 균등 분포(uniform distribution)를 이루도록 하였다.

6.2 실험 결과

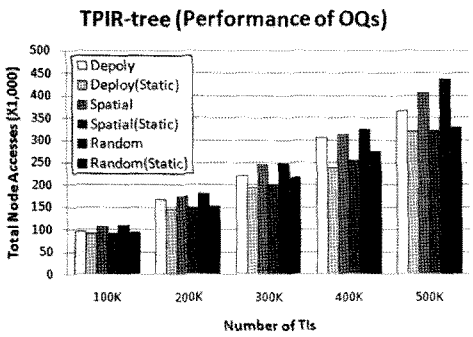
색인의 LID 도메인을 위해 초기 LID 정렬 방법을 세 가지 방법 - ① *PR200*에서 판독점의 배치에 따른 사전적인 LID 부여 방법(Deploy), ② *PR200*의 인식공간 간에 미리 정의된 공간 거리에 따른 LID 부여 방법(Spatial), ③ 인식공간으로의 랜덤 LID 부여 방법(Random) - 을 사용하였다. 세가지 초기 LID 정렬에 대해서 이 논문에서 제시하는 정적 LID 근접성과 동적 LID 근접성을 각각 적용하여 재정렬한 LID 순서 리스트를 생성하였다. 정적 LID 근접성을 적용한 정렬을 각각 Deploy(Static), Spatial(Static), Random(Static), 동적 LID 근접성을 적용한 정렬을 각각 Deploy(Dynamic), Spatial(Dynamic), Random(Dynamic)이라 표기한다. 또한, 두 LID 근접성의 동시 적용 효과를 확인하기 위해 정적 LID 근접성에 따른 LID 정렬 후 동적 LID 근접성에 의해 다시 정렬된 LID 순서 리스트를 생성하였으며, 이를 각각 Deploy(Static&Dynamic), Spatial(Static&Dynamic), Random(Static&Dynamic)이라 표기한다.

6.2.1 정적 LID 근접성 성능 비교

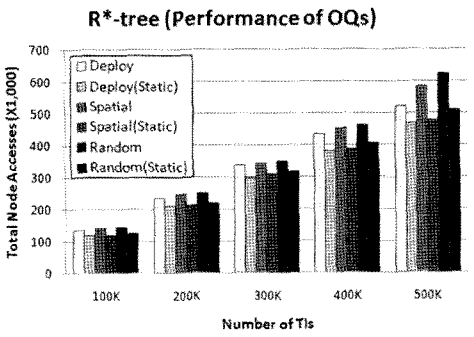
태그 위치추적 질의 성능 중 먼저 정적 LID 근접성에 따라 LID를 재정렬했을 때의 성능을 비교하였다. 이 실험에서는 초기 LID 순서 리스트인 Deploy, Spatial, Random과 정적 LID 근접성만을 적용한 Deploy(Static), Spatial(Static), Random(Static)을 사용한다. 6개의 LID 순서 리스트를 색인의 도메인에 적용 후 각 색인에 관측 질의와 궤적 질의를 각각 1,000번씩 수행하였다. 실험에 사용된 질의는 과거 및 현재 위치에 대한 질의를 모두 포함한다.

그림 11은 초기 LID 정렬과 정적 LID 근접성을 이용하여 정렬한 LID 순서 리스트에 대해서 각 색인에서의 태그 위치추적 질의의 성능을 보여주고 있다. 실험 결과 그래프에서 x축은 색인에 입력된 태그간격의 개수로 표 2에서 나열한 데이터 셋의 종류를 나타낸다. y축은 색인을 통한 1,000번의 관측 질의 또는 궤적 질의 수행 시 전체 노드 접근 횟수를 나타낸다. 실험 결과에서 살펴볼 수 있듯이 6개의 모든 LID 순서 리스트에 대해서 모든 데이터 셋에서 초기 LID 정렬보다 정적 LID 근접성을 적용한 LID 정렬이 좋은 질의 성능을 보여줌을 알 수 있다. 초기 100K 데이터 셋보다 태그간격의 입력 개수가 증가되는 500K 데이터 셋으로 갈수록 노드 접근 횟수의 차이가 벌어지며, 노드 접근 횟수의 성능 향상 비율도 태그간격이 증가할수록 조금씩 커짐을 알 수 있다. 특히, 다른 LID 정렬보다 Random(Static)과 Random의 성능 차이가 더 크게 나타났다. 이는 흐름경로 수가 많은 인식공간 사이를 태그가 이동할 확률이 높으므로 흐름경로 수를 기반으로 한 정적 LID 근접성이 태그간격의 개수가 증가할수록 질의의 성능에 긍정적인 영향을 주었기 때문이다.

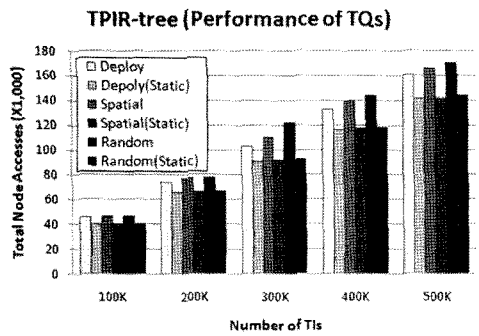
궤적 질의의 경우 흐름경로를 따라 이동하는 태그의 이동을 추적한다. 인식공간 사이 흐름경로의 수가 많은 경우 그렇지 못한 경우보다 많은 양의 태그가 이동할 확률이 높아진다. 따라서, 궤적 질의 시 해당 인식공간을 이동하는 태그의 이동 이력을 자주 검색할 것이다. 그래프에서 볼 수 있듯이 흐름경로 수를 기반으로 한



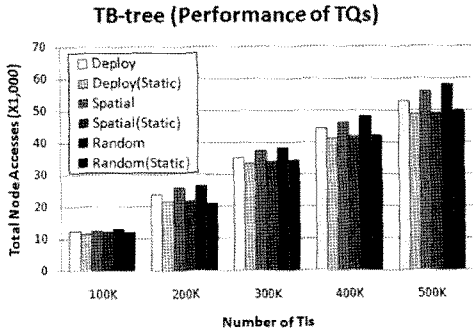
(a) 관측 질의 수행 시 노드 접근 횟수(TPIR-tree)



(b) 관측 질의 수행 시 노드 접근 횟수(R*-trce)



(c) 궤적 질의 수행 시 노드 접근 횟수(TPIR-tree)



(d) 궤적 질의 수행 시 노드 접근 횟수(TB-tree)

그림 11 정적 LID 근접성의 질의 성능 비교

정적 LID 근접성은 태그 이동을 추적하는 궤적 질의의 특징을 잘 반영하고 있다.

실험 결과를 통하여 Deploy(Static), Spatial(Static), Random(Static)을 적용한 색인의 성능이 각 데이터 셋에서 대체적으로 비슷한 검색 성능을 보여줌을 확인할 수 있다. GA가 비록 최적의 LID 순서 리스트를 생성하지는 못하더라도 최적 LID 순서 리스트에 근접한 결과를 생성하므로 Deploy(Static), Spatial(Static), Random (Static)에서 생성한 LID 순서 리스트는 LID 도메인에서 비슷한 LID 근접성으로 정렬되어 있다고 할 수 있다. 결과적으로, 이 논문에서 제시하는 정적 LID 근접성을 이용해 기존의 방법으로 인식공간에 부여된 LID를 재정렬함으로써 색인의 LID 도메인에 초기 LID 순서 리스트로 사용하는 것이 검색 성능 향상에 도움이 된다는 것을 알 수 있다.

6.2.2 동적 LID 근접성 성능 비교

이 절에서는 동적 LID 근접성을 사용하여 색인을 구축하였을 때 색인의 검색 성능을 비교한다. 앞절의 성능 비교에서 LID 순서 리스트 중 Deploy(Static)에 대부분의 색인 및 데이터 셋에서 가장 좋은 성능을 보였기 때문에, 동적 LID 근접성 성능 비교 시 Deploy(Static)을 초기 LID 순서 리스트로 선택을 한다.

동적 LID 근접성은 정적 LID 근접성과는 달리 수식 2에서 수식 9까지에서 정의한 시간매개변수 값을 반영하여 근접성을 측정해야 한다. 이를 위하여 100K에서 500K의 데이터 셋을 색인에 저장하는 동시에 태그 위치추적 질의를 반복적으로 수행한다. 데이터 셋 입력이 완료된 후 측정된 변수 값들을 기반으로 각 데이터 셋에 대한 동적 LID 근접성을 측정하고 LID 순서 리스트를 생성한다. 생성되는 LID 순서 리스트는 초기 LID 정렬로 Deploy를 사용한 후 생성한 Deploy(Dynamic)과 Deploy(Static)을 초기 LID 정렬로 사용한 후 생성한 Deploy(Static&Dynamic)이다.

동적 LID 근접성은 관측 질의에 대한 LID 근접성(LIDProx_OQ)과 궤적 질의에 대한 LID 근접성(LID-Prox_TQ)로 나누어진다. 각 질의타입이 동적 LID 근접성에 미치는 영향을 살펴보기 위하여 동적 LID 근접성 성능 비교의 첫번째 실험으로 관측 질의, 궤적 질의 중 하나의 질의 타입만 실행되는 환경을 대상으로 한다. 각 질의 타입에 대한 동적 LID 근접성을 측정하기 위해

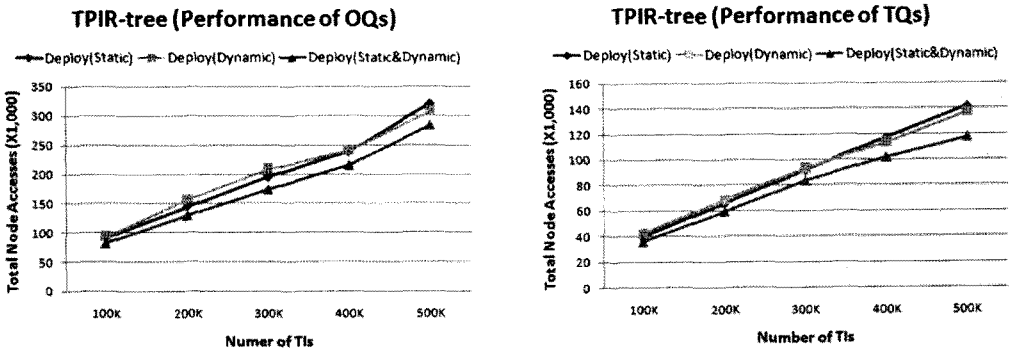
LID 재순서화 전 관측 질의 성능 측정 시는 10,000번의 관측 질의를 궤적 질의 성능 측정 시는 10,000번의 궤적 질의를 수행하였다.

그림 12는 TPIR-tree를 사용하였을 때 동적 LID 근접성 중 각각 LIDProx_OQ, LIDProx_TQ의 반영 효과를 측정하였다. 실험 결과 Deploy(Static&Dynamic)이 관측 질의와 궤적 질의 환경 모두에서 가장 성능이 우수하다. 이는 Deploy(Static)을 초기 LID 순서 리스트로 사용함으로써 흐름경로 수를 기반으로 한 정적 LID 근접성을 반영한 후에 태그간격이 입력되고 질의가 계속 수행되면서 발생되는 동적 속성들을 동적 LID 근접성을 통해서 반영을 했기 때문이다.

정적 LID 근접성에 대한 실험에서 Deploy는 Deploy(Static)에 비해서 성능이 좋지 않았으나 Deploy를 초기 LID 순서 리스트로 사용한 Deploy(Dynamic)은 태그간격이 증가함에 따라 Deploy(Static)과 비슷한 성능을 보여주고 있다. 500K의 데이터 셋에 대해서는 오히려 Deploy(Dynamic)이 Deploy(Static) 보다 성능이 더 우수하다. 이를 통하여, 연속적인 태그 이동과 이에 대한 태그 위치추적 질의가 계속해서 발생함으로써 동적 LID 근접성은 정적 LID 근접성보다 인식공간 간의 근접성을 좀 더 정확히 반영한다는 것을 알 수 있다.

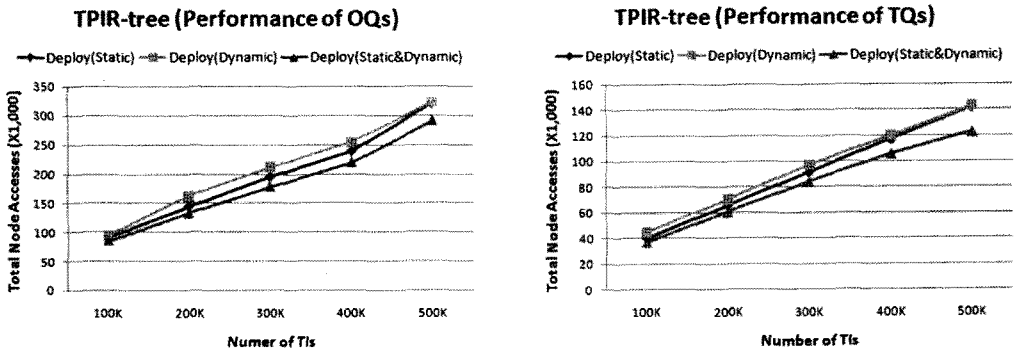
두번째로 관측 질의와 궤적 질의가 동시에 수행되는 환경에서 동적 LID 근접성에 대한 성능을 평가한다. 질의에 따라 변경되는 LIDProx_OQ, LIDProx_TQ 값을 모두 반영하기 위하여 각 데이터 셋에 대해서 100,000개의 태그간격이 삽입될 때마다 5,000번의 관측 질의와 5,000번의 궤적 질의를 LID 재순서화를 하기 전까지 계속해서 실행하였다.

그림 13은 이에 대한 질의 성능 비교 결과를 보여주고 있다. 실험 결과와 마찬가지로 Deploy(Static&Dynamic)의 성능이 가장 우수하며 Deploy(Dynamic)과 Deploy(Static)은 비슷한 성능을 보여주고 있다. 그러나, Deploy(Dynamic), Deploy(Static&Dynamic)은 그림 12의 실험결과와 비교했을 때 전 데이터 셋 구간에 대해서 질의 시 노드 접근 횟수가 증가했음을 알 수 있다.



(a) 관측 질의 시 노드 접근 횟수 (b) 궤적 질의 시 노드 접근 횟수

그림 12 각 질의타입에 대한 동적 LID 근접성의 질의 성능 비교



(a) 관측 질의 시 노드 접근 횟수 (b) 궤적 질의 시 노드 접근 횟수

그림 13 두 질의타입이 동시에 수행될 때 동적 LID 근접성의 질의 성능 비교

이는 관측 질의와 궤적 질의를 동시 수행함에 따라서 동적 LID 근접성 평가 시 LIDProx_OQ와 LIDProx_TQ가 각각 궤적 질의와 관측 질의에 대한 근접성에 부정적인 영향을 미쳤기 때문이다. 그 결과, Deploy(Dynamic)은 500K의 데이터 셋이 입력된 후에도 Deploy(Static)와 비슷한 성능을 보여주고 있다. 그럼에도 불구하고 Deploy(Static&Dynamic)은 전체 데이터 셋 구간에서 Deploy(Static) 보다 우수한 성능을 보여주고 있다.

7. 결론 및 향후 연구

이 논문에서는 RFID 시스템을 구성하는 위치에 대한 LID 부여가 태그 위치추적 질의의 특성을 고려하지 않았을 때 발생하는 문제점에 대해서 언급하였으며 그에 대한 해결책을 제시하고 있다. LID는 태그의 위치정보 제공자로서 TID, TIME과 함께 태그 위치추적 질의를 위한 색인에서 도메인으로 사용된다. 이에 따라서 태그 간격은 색인을 구성하는 도메인의 근접성 특성에 따라서 데이터 공간에서의 분포가 결정된다. 특히, 태그 위치추적 질의는 태그가 이동한 위치 이력을 검색하므로 세 가지 도메인 중 LID 도메인에서 LID의 정렬이 질의에 포함되는 태그간격의 분포에 결정적인 역할을 한다. 그러나, RFID 시스템을 구성하는 위치에 LID를 부여하는 기존의 방법들은 태그 위치추적 질의를 위한 태그간격의 분포를 고려하지 않기 때문에 질의 시 함께 접근되는 태그간격이 데이터 공간에서 근접해서 분포되지 않는다. 이로 인해 태그 위치추적 질의 수행 시 검색 성능을 떨어뜨리게 된다.

이 논문에서는 태그 위치추적 질의의 성능 향상을 위하여 질의에 포함되는 태그간격이 색인에서 근접해서 저장될 수 있도록 LID 도메인을 구성하는 LID를 재순서 기법을 제안하였다. 이를 위하여 이 논문에서는 LID 도메인에서 두 LID가 얼마나 가까이 정렬될 수 있는지를 결정하는 요소로써 태그의 이동이 가능한 지점의 개수를 기반으로 계산되는 정적 LID 근접성 함수와 시간에 따른 태그 위치추적 질의 및 태그 이동 특성을 통해서 계산되는 동적 LID 근접성 함수를 정의하였다. 또한, LID를 재정렬하기 위하여 각 LID 간의 LID 근접성을 이용하여 가중치 그래프를 구성하였으며 그래프의 순서화된 정점 집합을 생성하는 방법을 제시하였다. 실험 결과 LID 재순서 기법을 통하여 생성된 LID 순서가 기존의 방식으로 부여된 LID 순서보다 태그 위치추적 질의를 위한 색인에 적용했을 때 월등히 좋은 성능을 보여줌을 확인하였다.

RFID 시스템은 동적으로 태그가 위치를 이동하는 환경이며 이 논문에서 제안한 LID 근접성은 동적 요소에 의해서 시간이 지남에 따라서 계속 변경되기 때문에 색인에 저장된 태그간격 역시 시간이 지남에 따라 근접성이 변경된다. 이를 반영하기 위하여 주기적 또는 비주기적인 LID 재순서화가 필요하며 그에 따라 효율적으로 색인을 재구성하는 문제는 향후 해결되어야 한다.

참 고 문 헌

[1] Q. Z. Sheng, X. Li, and S. Zeadally, "Enabling Next-Generation RFID Applications: Solutions and Challenges," IEEE Computer, Vol.41(9), pp. 21-28, 2008.

[2] C. H. Lee and C. W. Chung, "Efficient Storage Scheme and Query Processing for Supply Chain Management using RFID," ACM SIGMOD, pp. 291-302, 2008.

[3] F. Wang and P. Liu, "Temporal Management of RFID Data," International Conference on VLDB, pp. 1128-1139, 2005.

[4] R. Derakhshan, M. E. Orlowska, and Xue Li, "RFID Data Management: Challenges and Oppertunities," IEEE International Conference on RFID, pp. 175-182, 2007.

[5] EPCglobal, http://www.epcglobalinc.org

[6] M. Harrison, "EPC Information Service - Data Model and Queries," Technical Report, Auto-ID Center, 2003.

[7] EPCglobal, "EPC Information Services (EPCIS) Specification," Version 1.0, EPCglobal Inc., 2006.

[8] C. H. Ban, B. H. Hong, and D. H. Kim, "Time Parameterized Interval R-tree for Tracing Tags in RFID Systems," International Conference on DEXA, pp. 503-513, 2005.

[9] Y. Theodoridis, M. Vazirgiannis, and T. Sellis, "Spatio-Temporal Indexing for Large Multimedia Applications," International Conference on Multimedia Computing and Systems, pp. 441-448, 1996.

[10] D. Pfoser, C. S. Jensen, and Y. Theodoridis, "Novel Approaches to the Indexing of Moving Object Trajectories," International Conference on VLDB, pp. 395-406, 2000.

[11] EPCglobal, "The EPCglobal Architecture Framework," EPCglobal Inc., 2005.

[12] EPCglobal, "EPCTM Radio-Frequency Identity Protocols Class-1 Generation-2 UHF RFID Protocol for Communications at 860 MHz-960MHz," Version 1.1.0, EPCglobal Inc., 2005.

[13] EPCglobal, "Reader Protocol (RP) Standard," Version 1.1, Ratified Standard, EPCglobal Inc., 2006.

[14] EPCglobal, "The Application Level Events (ALE) Specification," Version 1.1, EPCglobal Inc., 2008.

[15] T. Brinkhoff and H.-P. Kriegel, "The Impact of Global Clustering on Spatial Database Systems," International Conference on VLDB, pp. 168-179, 1994.

[16] I. Kamel and C. Faloutsos, "Hilbert R-tree: An

Improved R-tree using Fractals," International Conference on VLDB Conference, pp. 500-509, 1994.
[17] I. Kamel and C. Faloutsos, "On Packing R-trees," CIKM, pp. 490-499, 1993.
[18] D. S. Cho and B. H. Hong, "Optimal Page Ordering for Region Queries in Static Spatial Databases," International Conference on DEXA, pp. 366-375, 2000.
[19] A. Hutflesz, H.-W. Six, and P. Widmayer, "Globally Order Preserving Multidimensional Linear Hashing," ICDE, pp. 572-579, 1988.
[20] H. V. Jagadish, "Linear Clustering of Objects with Multiple Attributes," ACM SIGMOD, pp. 332-342, 1990.
[21] S. S. Skiena, "The Algorithm Design Manual," Springer-Verlag, New York Berlin Heidelberg, 1998.
[22] N. Beckmann and H. P. Kriegel, "The R*-tree: An Efficient and Robust Access Method for Points and Rectangles," ACM SIGMOD, pp. 322-331, 1990.
[23] D. Whitley, "A Genetic Algorithm Tutorial," Statistics and Computing, Vol.4, pp. 65-85, 1994.
[24] K. Marthias and D. Whitley, "Genetic Operators, the Fitness Landscape and the Traveling Salesman Problem," Parallel Problem Solving from Nature, pp. 219-228, 1992.
[25] D. Whitley and J. Dzebera, "Advance Correlation Analysis of Operators for the Traveling Salesman Problems," Parallel Problem Solving from Nature, pp. 68-77, 1994.
[26] Y. Theodoridis, J. R. O. Silva, and M. A. Nascimento, "On the Generation of Spatiotemporal Datasets, " International Symposium on Spatial Databases, pp. 147-164, 1999.
[27] T. Brinkhoff, "A Framework for Generating Network-Based Moving Objects," GeoInformatica, Vol.6(2), pp. 153-180, 2002.

안 성 우
1999년 부산대학교 컴퓨터공학과 졸업(공학사). 2001년 부산대학교 대학원 컴퓨터공학과 졸업(공학석사). 2009년 부산대학교 대학원 컴퓨터공학과 졸업(공학박사). 2009년~현재 부산대학교 U-Port 정보기술산학공동사업단 박사후 연수연구원. 관심분야는 LBS, 이동체 색인, 유비쿼터스 시스템, RFID 미들웨어

홍 봉 희
정보과학회논문지 : 데이타베이스
제 36 권 제 2 호 참조