論文
4-4-21

이미지 시퀀스 데이터베이스에서 우선순위 큐와 접미어 트리를 이용한 효율적인 유사 서브시퀀스 검색의 설계 (A Design for Efficient Similar Subsequence Search with a Priority Queue and Suffix Tree in Image Sequence Databases)

김 인 범(In-Bum Kim)[1]

요 약

본 논문은 우선순위 큐와 접미어 트리로 색인 구조를 생성한 후, 이미지 시퀀스 데이터베이스에서 다차원 타임 워핑 거리 함수를 이용하여 유사한 이미지 서브시퀀스를 신속하고 정확하게 검색할 수 있는 방법을 제안한다. 본 논문에서 제안된 방법은 사전에 정의된 중요도에 따라 선별된 이미지 시퀀스로 구성된 우선순위 큐 색인의 이미지 서브시퀀스에 대한 유사성 거리 계산을 첫 단계로 시행하여 유사한 서브시퀀스집합을 얻고 만족할 결과를 얻지 못했을 경우에는 두 번째 단계로 나머지 유사 서브시퀀스에 대해 디스크 기반의 접미어 트리를 색인 구조체로 하여 유사한 서브시퀀스를 검색하는 것이다. 하한 거리 함수를 활용하여 질의 이미지 시퀀스와 유사한 이미지 서브시퀀스를 검색하는 과정에서 생성 가능한 오류를 방지하면서 동시에 비 유사 이미지 서브시퀀스를 제거하도록 한다.

ABSTRACT

This paper proposes a design for efficient and accurate retrieval of similar image subsequences using the multi-dimensional time warping distance as similarity evaluation tool in image sequence database after building of two indexing structures implemented with priority queue and suffix tree respectively. Receiving query image sequence, at first step, the proposed method searches the candidate set of similar image subsequences in priory queue index structure. If it can not get satisfied results, it retrieves another candidate set in suffix tree index structure at second step. The using of the low-bound distance function can remove the dissimilar subsequence without false dismissals during similarity evaluating process between query image sequence and stored sequences in two index structures.

키워드 : 색인(Index), 우선순위 큐(Priority Queue), 접미어 트리(Suffix Tree), 유사 검색(Similarity Search), 이미지 시퀀스 데이터베이스(Image Sequence Database)

1) 정회원 : 김포대학 컴퓨터 계열 조교수

논문심사 : 2003. 5. 7.
심사완료 : 2003. 7. 7.

1. 서론

유사 검색(similarity search)이란 사용자에 의해 입력된 질의 시퀀스(query sequence)와 비슷한 패턴을 가지고 있는 시퀀스나 서브시퀀스를 찾는 행위이다. 그러나 시퀀스의 유사한 정도를 수치적으로 계산해서 판단하고 결정하는 것은 쉽지 않은 작업이다. 비록 어떤 두 시퀀스가 정성적(qualitatively)으로 매우 흡사할지라도 정량적(quantitatively)으로는 차이가 매우 클 수 있기 때문이다. 기존의 단순한 유사 검색 방법은 기본적으로 각 이미지 시퀀스나 서브시퀀스를 데이터베이스로부터 순차적으로 읽은 후 그것과 질의 이미지 시퀀스와의 거리를 계산하여 유사성을 판단한다. 비록 이러한 방법은 간단히 계산하여 그 결과를 얻을 수 있지만 데이터베이스의 범위와 크기를 확대해서 적용한다면, 심각한 성능 저하를 초래할 가능성이 있다. 따라서 유사 검색의 대상 데이터베이스를 확대하여 실용화하기 위해서는 보다 효과적인 색인 방법이 요구된다고 하겠다.

유사 검색은 전체 매칭과 서브시퀀스 매칭으로 크게 구분할 수 있다[1]. 전체 매칭이란 데이터 시퀀스와 질의 시퀀스가 같은 길이인 경우에 해당 질의 시퀀스와 유사한 데이터 시퀀스를 추출하는 것이고 서브시퀀스 매칭이란 질의 시퀀스와 유사한 서브시퀀스를 데이터 시퀀스 안에서 검색하는 것이다. 이러한 서브시퀀스 매칭의 경우에는 질의 시퀀스의 길이에 제약을 두지 않는다.

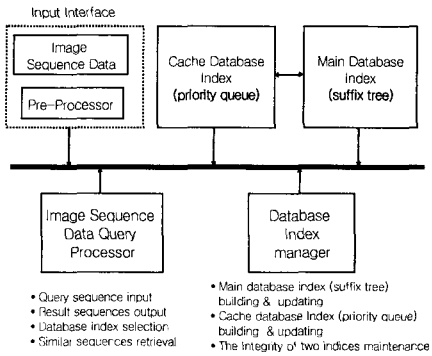
일차원 시퀀스를 대상으로 한 유사검색에 관한 연구는 크게 전체 시퀀스 매칭과 서브시퀀스 매칭으로 구분된다. 전체 시퀀스 매칭의 대표적인 연구로는 F-Index가 있는데[1], 이것은 먼저 시퀀스들을 이산 푸리에 변환(Discrete Fourier Transform)을 이용해 다차원 상의 점들로 변환한 후 이것을 R*-tree에 저장하고 질의 시퀀스가 입력되었을 때, 이것을 다차원 상의 점으로 변환한 후 이 점으로부터 일정한 거리 이내에 위치하는 점들을 찾아서 후보 집합에 추가시킨다. 서브시퀀스 매칭에 F-Index를 적용한 연구도 있었다[7].

유사성의 판단 기준으로 단순한 유클리드 거리(euclidean distance)를 사용한다면 잘못된 누락(false dismissal)이란 현상이 발생할 수 있다[1,2]. 이를 해결하기 위해서 최근의 연구는 스케일링, 쉬프팅, 정규화, 타임 워핑과 같은 다양한 시퀀스의 변형을 시도하고 있다[2,4,5,6,8,9,13]. 이 가운데서 관심을 끄는 것이 타임 워핑(time warping) 기법이다[4,15]. 이 타임 워핑은 추가 비용을 투자하지 않고 필요한 만큼의 자기 복제를 시행하는 변환이다. 타임 워핑에 의해 변형된 두 개의 시퀀스 사이의 최소 거리를 타임 워핑 거리라고 정의한다. 유클리드 거리는 비교되는 두 시퀀스의 길이가 같아야 유용하지만, 타임 워핑 거리는 임의의 길이의 어떠한 두 시퀀스에도 적용될 수 있다. 그러므로 시퀀스들의 길이가 다르거나 시퀀스 원소들이 서로 다른 시간 간격을 가지는 이미지 시퀀스 데이터베이스에서는 타임 워핑 거리를 사용하는 것이 더 적합하다고 할 수 있다. 각 시퀀스 원소들이 한 개의 수치 값을 가진다는 가정 하에, 타임 워핑 거리를 유사성 판단 기준으로 사용한 효율적인 서브시퀀스 매칭 방법이 연구되었는데[12], 이 방법은 색인 구조로 접미어 트리(suffix tree)를 적용하였고[19], 삼각 부등식을 가정하지 않았다. 또한 각 원소 값에 이산화 과정(discretization)을 도입하여 색인의 크기를 최소화하였고 유사 서브시퀀스를 빠뜨림 없이 추출하기 위해 접미어 트리를 순회하면서 하한 거리 함수를 필터링 도구로 사용하였다.

본 논문은 유사한 이미지의 서브시퀀스 검색 문제에 접미어 트리 색인구조를 기반으로 하는 우선순위 큐(priority queue)와 접미어 트리 색인 구조체를 도입하여 효율적인 서브시퀀스 검색이 가능하게 하는 색인방법을 제안한다. 이미지 시퀀스의 원소는 다중 값을 갖게 하고 거리 함수로는 다차원 타임 워핑 거리 함수를 사용하며 기본 색인 구조는 접미어 트리를 적용한다. 색인의 크기를 줄이고 질의 처리 성능을 높이기 위해 이산화 기법을 활용한다. 다차원 타임 워핑 거리 함수를 상대적 가중치를 고려해서 정의하고 다차원 원소

를 하나의 심볼로 이산화 시키는 것과 잘못된 누락 없이 비 유사 서브시퀀스들을 제거하기 위해 하한 다차원 타임 워핑 거리 함수를 정의한 것이 본 논문에서 제안된 주요내용이다. 본 논문의 효과적인 서브시퀀스 검색 방법은 단순히 질의 가능한 이미지 서브시퀀스들의 색인 저장 및 저장된 색인 정보를 활용한 질의 및 검색에만 제한을 둔 것이 아니라, 질의에 대한 결과를 얻기 위해 검색 수행 시 전문가의 지식, 수학적인 확률 정보 혹은 기타 검색에 유용한 정보를 활용하기 위해 일종의 캐쉬(cache) 기법을 모델링한 우선순위 큐로 구현된 색인 구조를 추가로 도입하여 검색 성능을 높인 것이다. 검색 대상의 우선순위를 잘 정의된 규칙에 따라 선정한 일부 서브시퀀스 집합으로 구성한 후, 미래의 사용자들에 의해 입력된 질의 시퀀스에 대해서 높은 우선 순위를 가진 시퀀스 집합을 대상으로 유사성을 계산한다. 만약에 이러한 시도에서 원하는 결과를 얻지 못하게 되면 나머지 유사 시퀀스 집합을 얻기 위해 접미어 트리의 색인 구조에 저장된 시퀀스들에 대해 유사성을 계산하며, 우선순위 큐와 접미어 트를 순회하여 얻은 유사 서브시퀀스의 합집합을 유사한 서브시퀀스 집합으로 결과를 반환한다. 질의 시퀀스와의 저장된 시퀀스와의 유사성 계산은 다차원 타임 워핑 거리 함수를 사용한다.

[그림 1]은 우선순위 큐 색인과 접미어 트리 색인을 이용한 이미지 시퀀스 데이터베이스에서의 효율적인 유사 서브시퀀스 검색을 위한 구성 요소를 보인다. 이미지 시퀀스 데이터 질의 프로세서(image sequence data query processor)는 이미지 서브시퀀스 질의 입력, 검색 작업에 사용할 색인 구조체 선택, 검색의 실행 및 검색 결과의 출력을 담당한다. 초기 이미지 시퀀스(raw image sequence)의 이산화 및 심볼화 작업등 전처리(pre-processing) 작업은 입력 인터페이스(input interface) 모듈에서 처리한다. 데이터베이스 색인 관리자(database index manager)의 기능은 접미어 트리(suffix tree)로 구성된 메인 이미지 시퀀스 데이터베이스 색인(main database index)과 우선순위 큐(priority queue)로 구성된 캐쉬 이미지 시퀀스 데이터베이스 색인(cache database index) 구조체를 생성하고 변경된 우선순위 정보에 따른 두 색인 구조체를 수정하고 이에 따르는 두 색인 구조체 사이의 무결성(integrity) 유지 및 제어작업을 한다.



[그림 1] 유사 이미지 서브시퀀스 검색의 주요 구성 요소

[Fig. 1] Components of similar image sub- sequence search system

2. 거리함수

어떤 두 시퀀스들의 유사한 정도를 판단하는 것은 일반적으로 쉽지 않는데 그 이유는 비록 정성적으로 유사한 두 시퀀스라고 할 지라도, 이것을 정량적으로 변환하여 표현하면 그 차이가 매우 클 수 있기 때문이다. 또한 비교되는 두 시퀀스들은 길이가 서로 다르거나 시퀀스들의 샘플링 비율이 다른 경우에는, 시퀀스들을 색인 공간에 매핑하고 유사 정도의 결정 문제에 유클리드 거리를 적용하는 것은 어렵거나 불가능하고 교차-상관관계(cross-correlation)와 같은 유사성 계산 기법을 사용할 수 없다. 이와 같은 문제가 발생 가능한 여러 응용에서는 이러한 문제를 타임 워핑 거리를 적용해서 해결하는 경우가 있다[4,15].

타임 워핑은 이산 값의 시퀀스와 연속 값의 시퀀스를 상호 비교하기 위한 전통적인 알고리즘을

일반화한 것으로, 두 시퀀스간의 차이의 최소 값을 얻기 위해, 한 시퀀스의 각 원소들이 다른 시퀀스의 하나 이상의 이웃된 원소들과의 매핑을 허용한다[15]. 임의의 두 시퀀스 X, Y를 대상으로, 타임 워핑 거리 D_{tw}는 다음 식과 같이 계산될 수 있다.

$$D_{tw}(\langle\rangle,\langle\rangle)=0$$
$$D_{tw}(X,\langle\rangle)=D_{tw}(\langle\rangle,Y)=\infty$$
$$D_{tw}(X,Y)=|X[1]-Y[1]|+\min\begin{cases}D_{tw}(X,Y[2:-])\\D_{tw}(X[2:-],Y)\\D_{tw}(X[2:-],Y[2:-])\end{cases}$$

위의 식에서, X는 n개의 원소들을 가지고 있는 일차원 시퀀스 $\langle X[1], \ldots, X[n]\rangle$를 의미하고, $X[i]$는 X의 i번째 원소, |X|는 X의 원소 개수를 표현한다. i번째부터 j번째까지의 원소들을 포함하는 X의 서브시퀀스는 $X[i:j]$로 표현된다. $X[i:-]$는 i번째 원소로부터 마지막 원소까지의 서브시퀀스를 의미한다. 여기서 $X[i:-]$는 i번째 원소로부터 시작하는 X의 접미어(suffix)를 구성한다. $\langle\rangle$는 시퀀스 내에 원소가 하나도 존재하지 않는 빈(empty) 시퀀스를 의미한다. $D_{tw}(X, Y)$는 순환 관계를 이용한 동적 프로그래밍(dynamic programming) 기법을 사용해서 계산이 가능하다[4]. 이러한 동적 프로그래밍 알고리즘은 그 실행 결과로 누적거리 테이블 T를 생성하는데, 최종 누적 거리 T[|X|,|Y|]는 X와 Y의 최소 거리이고 원소들의 매핑은 최소 누적 거리를 가지는 이전 셀 요소의 값을 선택하여 활용함으로써 누적거리 테이블 내에서 역으로 추적할 수 있다.

임의의 두 다차원 시퀀스 $\boldsymbol{X}$와 $\boldsymbol{Y}$를 대상으로 한 다차원 타임 워핑 거리 D_{mtw}는 아래 식을 이용해서 동적 프로그래밍 기법으로 계산할 수 있다.

$$D_{mtw}(\langle\rangle,\langle\rangle)=0$$
$$D_{mtw}(\boldsymbol{X},\langle\rangle)=D_{mtw}(\langle\rangle,\boldsymbol{Y})=\infty$$
$$D_{mtw}(\boldsymbol{X},\boldsymbol{Y})=D_{mbase}(\boldsymbol{X}[1],\boldsymbol{Y}[1])+\min\begin{cases}D_{mtw}(\boldsymbol{X},\boldsymbol{Y}[2:-])\\D_{mtw}(\boldsymbol{X}[2:-],\boldsymbol{Y})\\D_{mtw}(\boldsymbol{X}[2:-],\boldsymbol{Y}[2:-])\end{cases}$$

위의 식에서 다차원 시퀀스들은 굵은 글씨체로 표기하였다. 즉, $\boldsymbol{X}$는 n개의 원소들을 가지고 있는 다차원 시퀀스 $\langle\boldsymbol{X[1], \ldots, X[n]}\rangle$를 의미한다. 또한 $\langle\boldsymbol{X[i][1], \ldots, X[i][k]}\rangle$는 k개의 숫자 값을 가지는 $\boldsymbol{X}$의 i번째 원소를 의미하는데 이를 $\boldsymbol{X[i]}$로 표기한다. k개 특성을 가지는 임의의 두 다차원 원소 $\boldsymbol{X[i]}$와 $\boldsymbol{Y[j]}$에 대한 가중 거리 함수인 D_{mbase}는, h번째 특성의 가중치를 의미하는 W_h를 이용해서 다음과 같이 계산할 수 있다.

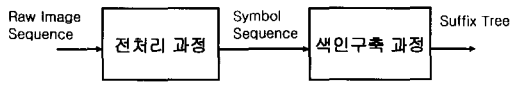
$$D_{mbase}(\boldsymbol{X}[i],\boldsymbol{Y}[j])=\sum_{h=1}^{k}W_h*|\boldsymbol{X}[i][h]-\boldsymbol{Y}[j][h]|$$

본 논문은 이미지 시퀀스 데이터베이스로부터 질의 시퀀스와 유사한 서브시퀀스를 효과적으로 검색하기 위해서 캐쉬개념을 모델링한 우선순위 큐와 접미어 트리 구조를 도입한 색인 방법을 제안한다. 이것은 앞에서 기술한 거리 함수를 활용해서 다음과 같이 기술할 수 있다. 즉, 이미지 질의 시퀀스 $\boldsymbol{q}$와 거리 허용 오차 $\boldsymbol{\varepsilon}$가 주어졌을 때, 가변 길이를 가지는 이미지 데이터 시퀀스들의 집합에서 다차원 타임 워핑 거리 $D_{mtw}(\boldsymbol{x}, \boldsymbol{q})$가 허용 오차 $\boldsymbol{\varepsilon}$ 이내인 조건을 만족하는 모든 이미지 서브시퀀스 x를, 이미지 시퀀스 데이터베이스 내에서 효과적으로 추출하는 방법을 제안하는 것이다.

3. 색인 생성

색인 생성단계에서는 초기 이미지 시퀀스(raw image sequence) 집합으로부터 이미지 데이터베이스 시퀀스 검색을 위한 색인(index)을 생성한다. [그림 2]와 같이, 색인 생성단계는 전처리(pre-processing) 과정과 색인 구축 과정으로 구분할 수 있다.

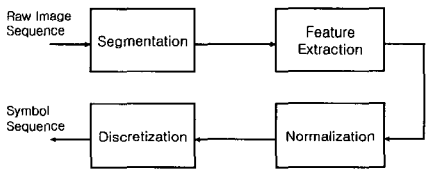
전처리 과정은 초기 이미지 시퀀스(raw image sequence) 집합을 입력받아 이를 처리하여 심볼 시퀀스(symbol sequence) 집합으로 변환시키고, 색인구축 과정에서는 심볼 시퀀스를 입력받아 접미어 트리 시퀀스를 생성하고 이것을 이진 병합(binary merge)하여 초기 디스크 기반의 접미어 트리 색인(suffix tree index)을 생성한다. 초기에 생성된 접미어 트리는 다음 단계에서 생성될 우선순위 큐 색인 구조체와 이차 접미어 트리 색인 구조체 형성의 기반이 된다.



[그림 2] 색인 생성단계의 구성
[Fig. 2] Index building processes

3.1. 전처리 과정

전처리 과정은 [그림 3]에 표현된 것과 같이 세그멘테이션(segmentation), 특성 추출(feature extraction), 정규화(normalization), 이산화(discretization)과정으로 구성된다. 초기 이미지 시퀀스(raw image sequence)들의 집합은 이러한 전처리 과정을 거쳐 심볼 시퀀스(symbol sequence) 집합으로 변환된다.



[그림 3] 전처리 과정의 구성
[Fig. 3] Preprocessing steps for index construction

세그멘테이션(segmentation)이란 배경 이미지로부터 관심 있는 객체들의 모양을 인식할 수 있는 경계를 구별하여 이를 추출하는 것이다. 특성 추출(feature extraction)은 세그멘테이션을 거쳐 인식된 객체로부터 다른 객체와 구별할 수 있는 여러 가지 특성 벡터를 계산하여 추출하는 것이다. 예를 들어, 이미지 시퀀스 데이터베이스에서, 특성벡터 A, B, C의 조합이 이미지 시퀀스를 구성하는 각 이미지를 구별할 수 있게 하는 특성벡터라고 가정하면, 이미지 시퀀스 데이터베이스내의 어떤 이미지 시퀀스 **X**는 〈(A[1], B[1], C[1]), ..., (A[n], B[n], C[n]))〉로 표시 가능하다. 여기서 n은 이미지 시퀀스 **X**를 구성하는 이미지의 수이다. 정규화 과정(normalization)은 특정한 차원의 상대적 가중치를 할당하거나 조절하는 것을 쉽게 하기 위해서 도입한다. 이러한 과정이 생략되면, 예를 들어 높은 평균값을 갖는 어떤 특성 A는 낮은 평균값을 갖는 다른 특성 C에 비해 시퀀스들의 유사성을 계산하는데 있어 상대적으로 많은 영향을 미치게 된다. 그러므로 모든 특성 차원이 유사성 결정에 균일하게 영향을 끼치도록 정규화 과정을 수행함으로 데이터 분포를 정규 분포(normal distribution)가 되도록 조절한다. 이산화 과정(discretization)의 목표는 색인 구조를 압축하여 저장 공간을 줄이고 질의 처리 성능을 향상시키기 위해서이다. 정규화 과정을 거친 다차원 원소들을 여러 개의 카테고리로 분류한 후 각각의 카테고리를 대표하는 고유한 심볼을 할당한다. 여러 가지 다차원 클러스터링 방법 중에서 본 논문에서는 다속성형 추상 계층분류법을 사용한다[22]. 이 방법은 확장 오류를 최소화시키기 위해 다차원 그룹의 경계 값을 결정하는 분류법으로 데이터의 값과 발생 빈도 분포를 동시에 고려하므로 비교적 정확한 결과를 생성할 수 있고 다른 방법에 비해 상대적으로 알고리즘이 간단하고 구현이 수월한 장점이 있다. k차원의 카테고리는 k개로 구성된 각 특성 벡터의 최대 값과 최소 값으로 표현된다.

3.2. 색인 구축 과정

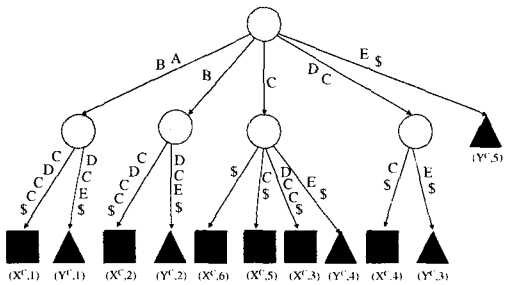
본 논문에서는 이미지 시퀀스 데이터베이스 내에서 효과적인 유사 서브시퀀스 검색을 위해 우선순위 큐와 접미어 트리를 색인 구조로 사용한다. 이 방법의 기본이 되는 골격은 접미어 트리이다. 초기에 생성되는 접미어 트리는 이미지 데이터베이스 시퀀스의 모든 접미어를 트리의 색인 구조 내에 저장한다. 우선순위 큐 색인 구조가 생성된 후에는 2차 접미어 트리 색인 구조에는 우선순위 큐에 저장된 시퀀스들을 제외한 나머지 모든 시퀀스를 저장하기 때문에 동일한 서브시퀀스에 대한 중복 유사성 계산은 배제된다. 접미어 트리는 어떤 시퀀스의 일련의 접미어를 모두 저장하고 있기 때문에 서브시퀀스 매칭에 아주 적합한 구조이고 트리의 구조가 삼각 부등식을 가정하고 있지 않기 때문에 타임 워핑 거리 함수를 이용해 유사성를

계산하여 결정하는 경우에도 잘못된 누락을 발생시키지 않는다.

접미어 트리에서 트라이(*trie*)는 키워드 집합에서 원하는 키워드를 신속하게 찾을 수 있는 색인 구조로, 접미어 트라이는 키워드 집합이 시퀀스의 접미어로 구성된 것이다[19]. 접미어 트라이를 구성하는 노드 중에서 하나의 자식만을 가지는 것들을 병합하면 접미어 트리가 얻어진다. 접미어 트리에서 각 시퀀스의 접미어는 단말 노드에 표현된다. 즉, X[i : -]는 단말노드에 (ID(X),i) 형태로 저장된다. X의 식별자는 ID(X)이고, i는 시퀀스 상에서 접미어가 시작되는 위치를 나타낸다. 예를 들어 [그림 4]의 X^C=〈A,B,C,D,C,C〉인 시퀀스에서 접미어 〈C,C〉가 저장되어 있는 단말노드는 (X^C,5)로 표현된다. 루트 노드 R과 단말 노드 L을 연결하는 경로 상의 모든 값들을 추적하면 해당 단말 노드L이 저장하고 있는 접미어를 얻을 수 있다. 루트 노드 R과 내부 노드 N을 연결하는 경로 상의 모든 값을 순서대로 추출하면 노드 N 아래에 위치하는 단말 노드들이 표현하는 접미어들의 공통 접두어(prefix)들 중에서 최대 길이의 접두어(prefix)를 얻을 수 있다.

다중 이미지 시퀀스로부터 접미어 트리를 생성하기 위해서, 본 논문에서는 디스크 기반의 점진적 접미어 트리 생성 기법[7]을 사용한다. 이 방법은 서로 다른 시퀀스 집합으로부터 생성된 두 개의 접미어 트리를 병합하기 위하여 두 개의 트리를 동시에 선위 순회하면서 공통된 서브시퀀스에 해당하는 경로를 결합하는 방법이다. 이 방법은 평균 길이가 $\overline{L}$인 m개의 데이터 시퀀스로부터 접미어 트리를 구성하기 위한 알고리즘의 복잡도는 O(m $\overline{L}$)로 계산된다.

[그림 4]는 초기 이미지 시퀀스(raw image *sequence*) X, Y로부터 변환된 두 개의 심볼 시퀀스, X^C=〈A,B,C,D,C,C〉, Y^C=〈A,B,D,C,E〉로부터 구성되어 병합된 접미어 트리를 나타내고 있다. 심볼 시퀀스 X^C로부터는 〈A,B,C,D,C,C〉, 〈B,C,D,C,C〉, 〈C,D,C,C〉, 〈D,C,C〉, 〈C,C〉, 〈C〉 등 6개의 접미어 시퀀스가 추출되고, 심볼 시퀀스 Y^C로부터는 〈A,B,D,C,E〉, 〈B,D,C,E〉, 〈D,C,E〉, 〈C,E〉, 〈E〉 등 5개의 접미어 시퀀스가 추출되어 서로 이진 병합되어 접미어 트리를 구성하고 있다. 접미어의 마지막에 위치하는 기호 $는 접미어 시퀀스의 마지막을 표현한다. [그림 4]에서 접미어 (X^C,4)인 시퀀스 〈D,C,C〉와 접미어 (Y^C,3)인 시퀀스 〈D,C,E〉는 공통의 최대 접두어로 시퀀스 〈D,C〉를 가지고 있다.



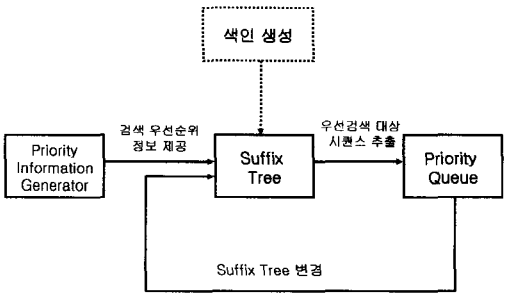
[그림 4] X^C=〈A,B,C,D,C,C〉, Y^C=〈A,B,D,C,E〉로부터 생성된 접미어 트리

[Fig. 4] A suffix tree constructed by merging X^C=〈A,B,C,D,C,C〉 and Y^C=〈A,B,D,C,E〉

4. 우선순위 큐 생성

[그림 5]에는 질의 이미지 시퀀스와 유사한 이미지 서브시퀀스를 검색할 때 첫 번째 검색 과정에서 이용되는 우선순위 큐 색인 구조를 생성하는 과정이 나타나 있다. 우선순위 정보 생성자(priority information generator)가 생성한 검색 우선순위 정보를 조건으로 이용해서 색인 생성과정에서 처음 생성한 초기 접미어 트리(suffix tree)를 순회하면서 해당 조건을 만족하는 서브시퀀스의 집합을 초기 접미어 트리로부터 추출한다. 추출된 서브시퀀스들은 자신들에게 부여된 우선순위 가중치를 기준으로 우선순위 큐를 구성하게 되고, 초기 접미어 트리는 우선순위 큐 색인 구조체를 형성하기 위해 추출된 서브시퀀스에 해당하는 접미어 시퀀스를 자신의 접미어 트리 구조체에서 제거한 후, 2차 접미어 트리 색인 구조체로 변경

된다. 이는 미래의 질의 이미지 시퀀스에 대한 유사한 시퀀스의 검색을 위해, 접미어 트리 색인 구조를 순회 시, 우선순위 큐의 시퀀스에 대해 중복하여 유사성 계산을 하지 않도록 하기 위함이다. 미리 정의된 우선순위 정책에 따라 제공되는 우선순위 정보가 어느 시점에서 변경되면 우선순위 큐와 2차 접미어 트리의 색인 구조체는 동적(Dynamic)으로 그 내용에 따라 변경되도록 한다. 이 작업은 [그림 1]에서 표현한 데이터베이스 색인 관리자(database index manager)가 처리하여 두 색인 구조체간의 정보의 무결성을 유지시킨다.

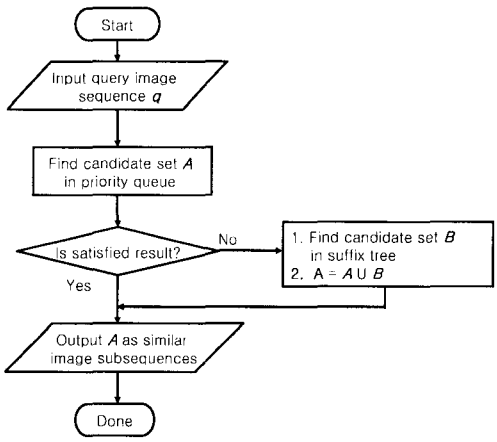


[그림 5] 우선순위 큐의 생성 및 접미어 트리의 수정
[Fig. 5] Processes of priority queue construction and suffix tree modification

5. 질의 처리

5.1. 질의 처리 과정

입력된 질의 이미지 시퀀스는 전처리 과정을 거쳐 다차원 시퀀스 q로 변환된다. 다음 단계는 [그림 6]에 기술되어 있는 것과 같이 질의 이미지 시퀀스와의 하한 거리가 허용 오차 ε 이내인 후보 서브시퀀스들을 추출하기 위해 먼저 우선순위 큐(priority queue)에서 질의 이미지 시퀀스와의 하한 거리가 허용 오차 이내인 후보 서브시퀀스를 찾는다. 첫 번째 검색단계에서 만족할 만한 결과를 얻지 못하면 접미어 트리(suffix tree) 색인 구조체를 루트 노드부터 순회하여 그 트리 상에 존재하는 시퀀스와의 거리를 계산하여 허용 오차 이내의 후보 서브시퀀스를 추출하고 그것을 기존의 후보 서브시퀀스집합에 추가하여 질의에 대한 결과로 반환한다.



[그림 6] 이미지 서브시퀀스 질의의 처리 과정
[Fig. 6] Processing steps for a query on image subsequence

필터링 작업에 하한 타임 워핑 거리를 사용함으로 인해서 실제 타임 워핑 거리가 허용오차 ε 보다 큰 서브시퀀스들이 추출된 후보 시퀀스 집합에 포함될 수 있다. 이러한 잘못된 서브시퀀스들이 후보 시퀀스 집합에 포함되는 것을 방지하기 위해 색인 검색을 종료한 후에, 적절한 후처리(post-processing) 작업을 수행하는 것이 필요하다. 즉, 이미지 시퀀스 데이터베이스에서 해당 서브시퀀스를 추출한 후 D_{mtw}를 사용해서 그들의 타임 워핑 거리를 계산하고, 최종적으로 질의 시퀀스와의 실제 타임 워핑 거리가 허용오차 ε 이내인 서브시퀀스들을 결과로 출력한다.

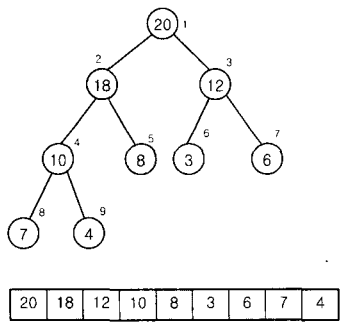
5.2. 색인 순회 알고리즘

본 논문에서 제안하는 색인 순회 과정은 [그림 6]에서 표현된 것 같이 우선순위 큐 색인 구조체에서의 색인 순회과정과 접미어 트리 색인 구조체에서의 색인 순회과정 등 두 단계로 구분된다.

우선순위 큐 색인 구조체에서의 질의 시퀀스와 유사한 서브시퀀스를 검색하는 과정이 알고리즘 1에 있다. 이 알고리즘은 우선 순위 큐가 최대 힙(max heap) 자료구조로 되어 있음을 전제로 한다. 최대 힙 자료구조는 완전 이진 트리(complete binary tree)로 간주될 수 있는 배열 개체(array object)이다. 트리의 각 노드에 저장되어 있는 값은 그 배열의 요소에 해당한다. 이 트리는 최하층을 제외하고는 왼쪽으로부터 오른쪽 방향으로 값을 채운다. 최대 힙(max heap) 구조는 루트 노드를 제외하고는 다음과 같은 특성을 가진다[20].

$$parent[i] = \lfloor i/2 \rfloor$$
$$X[\, parent[i] \,] \geq X[i]$$

즉, 노드 i의 부모 노드는 parent(i)라는 함수를 통해서 얻을 수 있고, 이러한 최대 힙 구조에서는 부모 노드의 값이 자신의 자식노드의 값보다는 작지 않은 특성을 보인다. 따라서 최대 힙 구조에서 가장 큰 값을 가지는 노드는 루트노드라고 할 수 있다. 참고로 노드 i의 왼쪽 자식(left child) 노드는 $2i$이고 오른쪽 자식(right child) 노드는 $2i$+1로 찾을 수 있다.



[그림 7] 최대 힙(Max Heap) 구조와 배열 객체 예
[Fig. 7] An example of max heap structure and its corresponding array object

[그림 7]에는 최대 힙 구조를 표현하는 이진 트리와 실제 배열 객체의 구조와 예를 보이고 있다. 각 노드를 나타내는 원 내부의 숫자는 우선순위 가중치를 의미하고 원 위의 숫자는 배열 객체 X에서의 위치 i를 표현한다. 최대 힙 구조의 특성을 예를 들어 설명하면 [그림 7]에서, i=5 인 경우에 parent[5] = $\lfloor$ 5/2 $\rfloor$ = 2 이므로 (X[parent[5]] = 18) >= (8 = X[5]) 인 부등식이 만족됨을 확인할 수 있다.

우선순위 큐 색인 구조체에서 유사 시퀀스를 검색하는 알고리즘 1에서 가장 큰 우선순위 값을 가지는 시퀀스는 최대 힙 구조에서 **ExtractMax()** 함수를 호출하여 그 결과를 얻음으로 결정할 수 있다. 알고리즘 2에는 **ExtractMax()**의 실행과정이 표현되어 있는데, 이 알고리즘의 실행시간은 최대 힙 구조에 저장되어 있는 시퀀스의 수를 n으로 가정했을 때, O(log$n$)이다. 루트 노드에서 X[1]을 선택하여 반환하는 과정에서 소요되는 시간은 O(1)이고, 최대 우선순위 가중치를 가지는 루트 노드를 우선순위 큐에서 제거하고 그 나머지 원소를 이용해서 최대 힙 구조로 재구성하는 **MaxHeapRebuld()** 모듈은 O(log$n$)의 실행시간을 요구한다. 따라서 **ExtractMax()**의 실행에 요구되는 시간은 O(logn)이 된다. 알고리즘 1에서 가장 높은 우선순위를 가지는 시퀀스 *aSequence*를 추출하여 앞에서 언급한 거리함수를 이용해서 질의 시퀀스와의 유사성을 계산하는 모듈 **CalDistance()**를 수행하고 이것의 반환 결과인 *dist*가 허용오차 ε 이내이면 *aSequence*를 후보집합 *candidateSet*에 포함시킨다.

우선순위 큐에 저장되어있는 시퀀스에 대한 전체 순회 대신에 유사성 계산을 위한 순회에 대하여 제한을 둘 수 있다. 이 경우에는 접미어 트리를 구성하는 시퀀스에 대한 추가적인 작업이 요구된다. 이러한 작업은 [그림 1]에 나타나 있는 데이터베이스 색인 관리자(database index manager)가 담당하게 된다. 우선순위 큐에 대한 유사성 계산을 시도할 시퀀스의 수를 k라고 가정할 때, 알고리즘 1의 실행시간은 O($k$log$n$)이 된다. 우선순위 큐 색인 구조체에 저장되어 있는 모든 시퀀스에 대한 전체 순회는 O($n$log$n$)의 실행시간을 요구하게 된다. 이 n은 전체 데이터베이스 시퀀스의 수에 비하면 상당히 작은 수가되어야 효

율적인 검색이 가능한데, 이를 위해서는 우선순위 정보 생성 및 관리에 있어서 어플리케이션 및 대상 데이터베이스에 대한 세밀한 연구와 정교한 설계가 필요하다.

알고리즘 3에는 접미어 트리 색인 구조체에서 실행되는 유사 서브시퀀스의 검색과정이 표현되어 있다. 루트(root)부터 검색을 시작하여 각 노드들을 깊이 우선(depth-first) 방식으로 방문한다. 노드 N의 i번째 자식 노드를 CN_i 라고 할 때, 노드 N을 방문하면서 새로운 후보를 찾기 위해 각각의 자식 노드 CN_i 조사한다. 이것은 루트로부터 CN_i 연결하는 경로가 새로운 후보에 포함될 수 있는 가를 확인하기 위해 $label(root, CN_i)$와 q 의 하한 타임 워핑 거리 D_{mtw-lb}을 구한다. 즉, 접미어 트리 상에는 초기 이미지 시퀀스 데이터(raw image sequence data)를 이산화 과정을 통해서 얻은 심볼 형태로 저장하고 있기 때문에, 접미어 트리 상에 존재하는 시퀀스와 질의 시퀀스 사이의 정확한 다차원 타임 워핑 거리를 얻을 수 없다. 따라서 다차원 타임 워핑 거리 D_{mtw}의 하한 거리를 계산해서 반환하는, 하한 타임 워핑 거리 $D_{mtw\text{-}lb}$를 도입한다. 이 다차원 타임 워핑 거리 $D_{mtw}(\boldsymbol{x}, \boldsymbol{y})$의 하한 거리 $D_{mtw-lb}(\boldsymbol{x}^C, \boldsymbol{y})$는 다음과 같이 계산할 수 있다.

$$D_{mtw-lb}(\langle\rangle, \langle\rangle) = 0$$

$$D_{mtw-lb}(\boldsymbol{x}^C, \langle\rangle) = D_{mtw-lb}(\langle\rangle, \boldsymbol{y}) = \infty$$

$$D_{mtw-lb}(\boldsymbol{x}^C, \boldsymbol{y}) = D_{mbase-lb}(\boldsymbol{x}^C[1], \boldsymbol{y}[1]) + \min \begin{cases} D_{mtw-lb}(\boldsymbol{x}^C, \boldsymbol{y}[2:-]) \\ D_{mtw-lb}(\boldsymbol{x}^C[2:-], \boldsymbol{y}) \\ D_{mtw-lb}(\boldsymbol{x}^C[2:-], \boldsymbol{y}[2:-]) \end{cases}$$

$$D_{mbase-lb}(C, \boldsymbol{y}[1]) = \sum_{h=1}^{k} W_h * D_{base-lb}(C[h], \boldsymbol{y}[1][h])$$

$$D_{base-lb}(C[h], \boldsymbol{y}[1][h]) = \begin{cases} 0 & \text{if } C[h].\min \le \boldsymbol{y}[1][h] \le C[h].\max \\ C[h].\min - \boldsymbol{y}[1][h] & \text{if } \boldsymbol{y}[1][h] < C[h].\min \\ \boldsymbol{y}[1][h] - C[h].\max & \text{if } \boldsymbol{y}[1][h] > C[h].\max \end{cases}$$

임의의 두 다차원 원소들에 대해 언제나 하한 거리 $D_{mbase-lb}$가 원래의 D_{mbase}보다 작거나 같은 값을 반환하므로, 당연히 두 다차원 시퀀스들에 대해 항상 D_{mtw-lb}가 D_{mtw}보다 크지 않는 값을 반환한다. 질의 처리 알고리즘은 비 유사 서브시퀀스를 제거하기 위해 D_{mtw-lb}를 사용한다. 따라서 다차원 이미지 데이터 시퀀스와 질의 이미지 시퀀스간의 다차원 타임 워핑 거리 D_{mtw}가 허용 오차 ε 이내일 때, 그들의 하한 타임 워핑 함수 거리 D_{mtw-lb}는 확실히 ε 이하이다. 이것은 질의 이미지 시퀀스로부터 허용오차 ε 보다 크지 않은 모든 데이터 시퀀스들이 후보 집합에 확실히 포함되도록 해준다.

알고리즘 1 : PriorityQueueTraversal Algorithm

Input : priority queue ***PQ***, query seqence ***q***, distance tolerance ε, retrieval number in priority queue database index ***k***

Output: candidateSet

```
candidateSet ← {}
for i ← 1 to k do
    aSequence = ExtractMax(PQ)
    dist ← CalDistance(aSequence, q)
  if dist ≤ ε then
    candidateSet ← candidateSet ∪ {aSequence}
loop
return candidateSet
```

알고리즘 2: ExtractMax Algorithm

Input : a set of image subsequence having heap structure ***X*** (i.e. a priority Queue)

Output: the sequence having max priority *max*

```
if size[X] < 1 then
    error "No Sequence"
max ← X[1]
X[1] ← X[size[X]]
size[X] ← size[X] - 1
MaxHeapRebuild(X, 1)
return max
```

알고리즘 3에서, $label(root, CN_i)$와 q사이의 하한 타임 워핑 거리를 계산하기 위해서 동적 프로그래밍 기법을 사용하는데, X축 상에는 q, Y축 상에는 $label(root, CN_i)$를 위치시킨 상태에서 누

적 거리 테이블을 작성하고 N이 루트라면, 누적 거리 테이블은 최하층(bottom)에서 만들고, 그 외의 경우에는, 함수 AddRow(T, q, $label(N, CN_i)$, D_{mtw-lb})를 이용하여, 루트부터 N까지 축적되어온 기존의 테이블 T위에 $label(N, CN_i)$에 해당하는 새로운 행(row)을 추가함으로써 누적 거리 테이블을 생성한다. 새로 추가되는 행의 가장 오른쪽 열의 값이 허용 거리 오차 ε 이하이면 $label(root, CN_i)$을 후보 집합에 추가시킨다. 실행과정에서 자기 자신을 호출하므로 CN_i가 조사된 후에는 CN_i의 자식 노드들이 방문된다. 그러나 탐색 공간을 줄이기 위해서는 모든 자식노드를 방문하는 것보다는 방문이 필요한 자식 노드만을 방문한다. 이를 위해 누적 거리 테이블의 마지막 행을 점검하여 그곳에 저장된 값 중에서 ε 보다 작거나 같은 것이 하나라도 있으면 CN_i의 자식 노드들을 방문한다. 그렇지 않으면 N의 다음 자식인 CN_{i+1}을 방문하여 수행을 계속한다.

알고리즘 3: IndexTraversal Algorithm

Input : node N, query sequence q, distance tolerance ε, cumulative distance table T

Output: candidateSet

```
candidateSet ← {};
CN ← GetChildren(N);
for i ← 1 to |CN| do
  CT_i ← AddRow(T, q, label(N, CN_i), D_mtw-lb);
  Let dist be the value in the rightmost column of newly added row;
  Lit minDist be the minimum value in the newly added row;
  if dist ≤ ε then
    candidateSet ← candidateSet ∪ {label(root, CN_i)};
  if minDist ≤ ε then
    candidateSet ← candidateSet ∪ IndexTraveral(CN_i, q, ε, CT_i);
loop
return candidateSet
```

이전 연구들의 기본 방법은 각 이미지 시퀀스를 데이터베이스로부터 읽어 이미지 시퀀스에 포함된 접미어의 수만큼 누적 거리 테이블을 만든다. 개별 이미지의 특성 차원의 수를 k라고 할 때, 질의 이미지 시퀀스 q와 길이가 L인 접미어에 대한 누적 거리 테이블을 생성하기 위해서는 $O(kL|q|)$의 계산이 필요로 한다. 평균 길이가 $\overline{L}$인 m개의 데이터 시퀀스 내에 존재하는 접미어의 수는 $m\overline{L}$개이고, 그것의 평균 길이는 $\frac{\overline{L}+1}{2}$이다. 따라서 이러한 방법을 사용하기 위해서는 $O(km\overline{L}^2|q|)$의 계산량이 요구된다.

본 논문에서 제안된 알고리즘은 이러한 방법보다 우수한 성능을 보인다. 그 이유는 우선순위 큐 색인 구조체를 전문가적 지식, 수학적 확률적인 정보, 혹은 다른 유용한 정보를 이용해서 신속한 검색을 위해 생성하였으므로 질의 이미지 서브시퀀스와 유사한 서브시퀀스들이 우선순위 큐 색인 구조체 내에 존재하면, 컴퓨터 메모리의 계층적(hierarchy) 구조에서의 캐쉬메모리(cache memory)를 액세스하는 것과 유사하게 효율적으로 후보 서브시퀀스 집합을 얻을 수 있고, 최악의 경우 우선순위 큐에 후보 시퀀스들이 전혀 존재하지 않은 경우일지라도, 접미어 트리의 순회를 통해 효율적으로 후보 시퀀스의 집합을 얻을 수 있기 때문이다. 이것은 접미어 트리 색인 구조체에서 검색대상이 우선순위 큐 색인 구조체에 존재하는 서브시퀀스는 제외되도록 설계되었고, 또한 접미어 트리 색인 구조체의 순회과정에서 사용되는, 제안된 가지치기(branch-pruning) 방법이 탐색 공간을 줄이고, 공통 접두어(prefix)를 가진 접미어들이 색인 순회 과정에서 누적 거리 테이블을 공유하기 때문이다.

6. 결론

본 논문에서는 입력된 질의 이미지 시퀀스와 유사한 서브시퀀스를 이미지 시퀀스 데이터베이스로부터 오류 없이 효과적으로 검색하기 위해 우선순위 큐와 디스크 기반의 접미어 트리로 구성된 두 단계

의 색인 구조체를 제안했다. 이것의 목적은 대용량 이미지 시퀀스 데이터베이스에서 정확하고도 신속한 유사 서브시퀀스의 검색을 목표로 한다. 이미지 서브시퀀스들은 길이가 다르거나 샘플링 비율이 다를 수 있기 때문에, 이를 해결하기 위해서 본 논문에서는 다차원 타임 워핑 거리를 유사성 기준으로 사용했다. 이러한 다차원 타임 워핑은 다차원 시퀀스들이 시간 축으로 확장되는 것이 가능함으로 다차원 이미지 시퀀스간의 거리를 최소화시키는 것이 가능하다. 색인의 크기를 줄이고 질의 처리 속도를 높이기 위해 다차원 이산화 과정을 도입하였으며 비 유사 서브시퀀스를 오류 없이 제거하기 위해 하한 거리 함수를 사용하였다.

이미지 시퀀스 데이터베이스는 이미지들의 순차 리스트로 구성된 이미지 시퀀스들의 집합이고 유사 검색은 주어진 질의 시퀀스와 비슷한 변화 패턴을 가지고 있는 시퀀스나 서브시퀀스를 찾는 동작으로 최근에 의학분야, 디지털 라이브러리, 영화, 음악 및 비디오, 데이터 웨어 하우징 등과 같은 많은 응용에서 그 중요성이 점차 강조되고 있다. 본 논문에서 제안된 방법의 활용 예를 들면, 의학 분야에서 어떤 환자의 현재 증상이나 특징을 보이는 이미지 서브시퀀스 데이터를 가지고, 이미지를 포함한 대용량 의학 데이터베이스에서 유사한 특징을 갖는 의학적 정보 및 이미지 시퀀스를 신속하고 정확하게 검색하여 이를 활용해서 환자 치료에 이용할 수 있을 것이다. 또한 본 논문에서 제안된 색인 방법과 거리 함수를 사용한다면 수많은 비디오나 사운드, 이미지 등 멀티미디어 데이터베이스부터 원하는 객체와 유사한 특성을 보이는 전체 혹은 일부 멀티미디어 객체를 정확하고 효율적으로 검색할 수 있을 것이다.

※ 참고문헌

[1] R. Agrawal, C. Faloutsos, and A. Swami. Efficient similarity search in sequence databases, In Proc. Int'l Conf. on Foundations of Data Organization and Algorithms(FODO), pp. 69-84, 1993.

[2] R. Agrawal, K. Lin, H. S. Sawhney, and K. Shim. Fast similarity search in the presence of noise, scaling, and translation in time-series databases. In Proc. Int'l Conf. on Very Large Data Bases (VLDB), pp. 490-501, 1995.

[3] R. Agrawal, G. Psaila, E. L. Wimmers, and M. Zait. Querying shapes of histories. In Proc. Int'l Conf. on Very Large Data Bases (VLDB), pp. 502-514, 1995.

[4] D. J. Berndt and J. Cliford. Finding patterns in time series: A dynamic programming approach. In Advances in Knowledge Discovery and Data Mining, pp. 229-248, AAAI/MIT, 1996.

[5] K. W. Chu and M. H. Wong. Fast time-series searching with scaling and shifting. In Proc. ACM Symposium on Principles of Database Systems (PODS), pp. 237-248, 1999.

[6] G. Das, D. Gunopulos, and H. Mannila. Finding similar time series. In Proc. Principles and Practice of Knowledge Discovery in Databases (PKDD), pp. 88-100, 1997.

[7] C. Faloutsos, M. Ranganathan, and Y. Manolopoulos. Fast subsequence matching in time-series databases. In Proc. ACM Int'l Conf. on Management of Data (SIGMOD), pp. 419-429, 1994.

[8] D. Q. Goldin and P. C. Kanellakis. On similarity queries for time-series data: Constraint specification and implementation. In Proc. Constraint Programming, pp. 137-153, 1995.

[9] S. W. Kim, S. Park, and W. W. Chu. An index-based approach for similarity search supporting time warping in large sequence databases. In Proc. IEEE Int'l

Conf. on Data Engineering (ICDE), pp. 607-614, 2001.

[10] R. Lienhart, C. Kuhmunch, and W. Effelsberg. On the detection and recognition of television commercials. In Proc. IEEE Int'l Conf. on Multimedia Computing and Systems, 1997.

[11] R. Mohan. Video sequence matching. In Proc. Int'l Conf. on Acoustics Speech and Signal Processing (ICASSP), 1998.

[12] S. Park, W. W. Chu, J. Yoon, and C. Hsu. Efficient searches for similar subsequences of different lengths in sequence databases. In Proc. IEEE Int'l Conf. on Data Engineering(ICDE), pp. 23-32, 2000.

[13] S. Park, S. W. Kim, J. S. Cho, and S. Padmanabhan. Prefix-querying: An approach for effective subsequence matching under time warping in sequence databases. In Proc. ACM Int'l Conf. on Information and Knowledge Management (CIKM), pp. 255-262, 2001.

[14] S. Park, D. Lee, and W. W. Chu. Fast retrieval of similar subsequences in long sequence databases. In Proc. IEEE Knowledge and Data Engineering Exchange Workshop (KDEX), pp. 60-67, 1999.

[15] L. Rabinar and B.-H. Juang. Fundamentals of Speech Recognition. Prentice Hall, 1993.

[16] J. M. Sanchez, X. Binefa, J. Vitria, and P. Radeva. Local color analysis for scene break detection applied to TV commercials recognition. In Proc. Visual 99, 1999.

[17] H. Shatkay and S. B. Zdonik. Approximate queries and representations for large data sequences. In Proc. IEEE Int'l Conf. on Data Engineering (ICDE), pp. 536-545, 1994.

[18] K. Shim, R. Srikant, and R. Agrawal. High-dimensional similarity joins. In Proc. IEEE Int'l Conf. on Data Engineering (ICDE), pp. 301-311, 1997.

[19] G. A. Stephen. String Searching Algorithms. World Scientific Publishing, 1994.

[20] Thomas H. Cormen, Charles E. Leiserson, Ronald L, Rivest and Clifford Stein, Introduction to Algorithms, MIT press, 2001

[21] A. Vailaya, W. Xiong, and A. K. Jain. Query by video clip. In Proc. Int'l Conf. on Pattern Recognition, 1998.

[22] W. W. Chu and K. Chiang. Abstraction of high level concepts from numerical values in databases. In Proc. AAAI Workshop on Knowledge Discovery in Databases, pp. 133-144, 1994.

김 인 범

1989, 1991년 서울대학교 컴퓨터 공학과(공학사) 공학 석사 대우통신 종합연구소 오라클 코리아 연구원

현재 김포대학 컴퓨터 계열 조교수

현재는 시공간 데이터베이스, 컴퓨터 이론 및 알고리즘, 컴퓨터 구조 및 네트워크, 컴퓨터 보안 분야 연구

관심 분야 : 멀티미디어 데이터베이스 시스템, 컴퓨터이론, 네트워크, 컴퓨터 보안