시계열 데이타베이스에서 임의 계수의 이동평균 변환을 지원하는 서브시퀀스 매칭 알고리즘 †

노 웅 기*°, 김 상 욱**, 황 규 영*, 심 규 석*
*한국과학기술원 전산학과, 첨단정보기술 연구센터 **강원대학교 정보통신공학과
{woong,kywhang,shim}@mozart.kaist.ac.kr wook@cc.kangwon.ac kr

A Subsequence Matching Algorithm Supporting Moving Average Transformation of Arbitrary Order in Time-Series Databases

Woong-Kee Loh*°, Sang-Wook Kim**, Kyu-Young Whang*, and Kyuseok Shim*
*Department of Computer Science and Advanced Information Technology Research Center (AITrc), Korea Advanced Institute of Science and Technology (KAIST)
**Department of Information and Telecommunication, Kangwon National University

요 약

본 논문에서는 시계열 데이타베이스에서 임의 계수의 이동평균 변환을 지원하는 서브시퀀스 매칭 알고리즘을 제안한다. 응용 분야와 분석하려고 하는 시계열 데이타의 특성에 따라 잡음의 영향을 줄이는 정도와 경향을 파악하는 주기가 달라지므로 이동평균 계수의 선택도 달라진다. 본 논문에서는 하나의 이동평균 계수에 대해서 생성한 인덱스만을 이용하여 인덱스가 생성되어 있지 않은 계수에 대해서도 탐색을 수행하는 방법을 제안한다. 이때, 제안된 탐색 기법이 질의 결과로 반환되어야 할 서브시퀀스를 모두 찾아내지 못하는 착오 기각이 발생하지 않음을 증명한다. 실험 결과, 모든 이동평균 계수에 대해 인덱스가 생성되어 있는 경우와 비교하여 탐색 성능의 저하는 42 % 이내였으며, 제안된 알고리즘의 탐색 성능이 순차 검색에 비하여 최대 2.7 배 우수하였다.

1 서 론

시계열(time series) 데이타는 일정한 시간 주기마다 얻어진 연속된 실수 값들로 이루어진 데이타이다 [2, 4]. 시계열 데이타는 새로운 데이타베이스 분야에서 점차 중요성이 더해가고 있으며, 시계열 데이타 간의 유사성 문제는 그러한 분야에서 가장 관심을 끄는 문제 중의 하나이다 [2]. 시계열 데이타베이스에 저장된 시계열 데이타를 **데이타 시퀀스***(data sequence)*라고 부르며, 질의 시퀀스와 유사한 데이타 시퀀스를 검색하는 연산을 **유사 시퀀스 매칭***(similar sequence matching)*이라고 한다 [1, 2, 4, 6].

본 논문에서는 임의 계수의 이동평균 변환 [3, 5, 6]을 효율적으로 지원하는 서브시퀀스 매칭 알고리즘에 관하여 논의한다. 이동평균 변환은 데이타 시퀀스의 연속되는 k 개 값의 평균 값을 순차적으로 나열하는 변환이다. 여기에서, k 값을 **이동평균 계수***(moving average order)* 또는 간단히 **계수***(order)*라 부른다. 이동평균 변환은 시계열 데이타 내의 잡음(noise)의 영향을 감소시킴으로써 시계열 데이타 전체의 경향을 파악하는 데에 유용하다 [3]. 응용 분야와 시계열 데이타의 특성에 따라 잡음의 영향을 줄이고자 하는 정도와 경향을 파악하고자 하는 주기가 달라지므로 각 응용 분야와 분석하고자 하는 시계열 데이타에 적합한 이동평균 계수를 적절하게 선택해야 한다 [5]. 따라서, 임의의 이동평균 계수를 지원해야 한다.

본 논문에서는 기존의 서브시퀀스 매칭 알고리즘 [4]을 확장하여 임의 계수의 이동평균 변환을 지원하는 서브시퀀스 매칭 알고리즘을 제안한다. 본 논문에서 제안하는 새로운 탐색 기법은 하나의 이동평균 계수 k에 대하여 생성된 인덱스만을 이용하여 임의의 계수 m ($\leq k$)에 대하여 이동평균 변환된 서브시퀀스 매칭을 수행한다. 이때, 제안된 탐색 기법에서 질의 결과로 반환되어야 할 서브시퀀스를 모두 찾아내지 못하는 착오 기각(false dismissal)이 발생하지 않음을 보인다. 실험을 통하여 하나의 이동평균 계수에 대한 인덱스만을 이용하더라도 모든 계수에 대해 인덱스가 존재하는 경우와 비교하여 탐색 성능이 크게 저하되지 않으며, 제안된 알고리즘이 순차 검색(sequential scan)에 비하여 탐색 성능이 우수함을 보인다.

2 문제 정의

정의 1 시퀀스 $\vec{X} = (x_i)$ $(0 \leq i < n)$를 이동평균 계수 k $(1 \leq k \leq n)$에 의해 이동평균 변환한 시퀀스 $\vec{X}_{(k)} = (x_{(k)j})$ $(0 \leq j < n-k+1)$는 다음과 같이 정의한다 [3, 5].

$$x_{(k)j} = \frac{1}{k}\overbrace{(x_j + \cdots + x_{j+k-1})}^{k} = \frac{1}{k}\sum_{i=j}^{j+k-1} x_i$$

여기에서, n은 데이타 시퀀스 $\vec{X}$의 길이이다. □

임의 계수의 이동평균 변환을 지원하는 서브시퀀스 매칭 문제는 다음과 같이 정의한다. 질의 시에 질의 시퀀스 $\vec{T}$와 함께 이동평균 계수 k가 함께 주어지면, 질의 시퀀스 $\vec{T}$를 k-이동평균 변환한 시퀀스 $\vec{T}_{(k)}$를 비교 대상으로 삼는다. 데이타베이스의 모든 데이타 시퀀스 $\vec{S}$에 대하여 $\vec{S}$를 k-이동평균 변환한 시퀀스 $\vec{S}_{(k)}$의 임의의 서브시퀀스 $\vec{X}_{(k)}$가 $\vec{T}_{(k)}$와 유사하면 데이타 시퀀스 $\vec{S}$와 $\vec{S}_{(k)}$ 내에서의 서브시퀀스 $\vec{X}_{(k)}$의 위치를 반환한다. 이때, 서브시퀀스 $\vec{X}_{(k)}$와 질

† 본 연구는 첨단정보기술 연구센터를 통하여 과학재단의 지원을 받았음.

의 시퀀스 $\vec{T}_{(k)}$의 길이는 같다.

3 임의 계수 이동평균 변환 서브시퀀스 매칭 기법

본 논문에서는 두 단계로 나누어 문제를 해결한다. 먼저, 첫번째 단계에서는 미리 정해진 하나의 이동평균 계수의 질의만을 지원하는 고정 계수 이동평균 변환 서브시퀀스 매칭 알고리즘을 제시하고, 두번째 단계에서는 이를 확장하여 임의 계수 이동평균 변환 서브시퀀스 매칭 알고리즘을 제시한다.

고정 계수 이동평균 변환 서브시퀀스 매칭을 위한 인덱싱 및 탐색은 참고문헌 [4]의 알고리즘을 단순히 응용한 것으로 다음과 같이 수행한다. 인덱싱 과정에서는 먼저 모든 데이타 시퀀스 $\vec{S}$의 k-이동평균 변환 시퀀스 $\vec{S}_{(k)}$를 생성한다. 생성된 시퀀스 $\vec{S}_{(k)}$를 일정한 길이 w_k의 슬라이딩 윈도우 $\vec{\chi}_{(k)i}$로 분할하고 DFT 변환하여 f ($< w_k$) 차원 인덱스에 저장한다. 탐색 과정에서는 먼저 질의 시퀀스 $\vec{T}$에 대하여 k-이동평균 변환 시퀀스 $\vec{T}_{(k)}$를 생성한다. 생성된 시퀀스 $\vec{T}_{(k)}$에 대하여 길이 w_k인 p 개의 윈도우 $\vec{\tau}_{(k)j}$ $(0 \le j < p)$로 분할한다. 각 윈도우 $\vec{\tau}_{(k)j}$에 대하여 다음의 공식 (1)을 만족하는 모든 윈도우 $\vec{\chi}_{(k)j}$로 후보 집합을 구성한다.

$$d\left(\vec{\chi}_{(k)j}, \vec{\tau}_{(k)j}\right) \le \frac{\epsilon}{\sqrt{p}} \tag{1}$$

여기에서, ϵ은 탐색 영역이다. 후보 집합에 포함된 모든 서브시퀀스에 대하여 직접 데이타베이스를 액세스하여 실제 거리를 측정함으로써 최종 적합성을 판정한다.

임의 계수 이동평균 변환 서브시퀀스 매칭의 인덱싱 과정에서는 미리 정해진 하나의 계수 k 값에 대해서 인덱싱을 수행한다. 인덱싱하지 않은 이동평균 계수 m에 대한 질의를 지원하기 위하여, 고정 계수 이동평균 변환 인덱싱 과정에서 생성한 인덱스에 저장된 각 슬라이딩 윈도우에 최소값/최대값 정보를 추가한다. 본 논문에서는 선택된 이동평균 계수 k 값에 대하여 생성한 인덱스를 k-인덱스*(k-index)*라 한다.

탐색 과정에서는 질의 시에 주어진 이동평균 계수 m에 따라 질의 시퀀스 $\vec{T}$를 m-이동평균 변환한 $\vec{T}_{(m)}$에 대하여 서브시퀀스 매칭을 수행해야 한다. 즉, 공식 (1)과 유사한 다음의 공식 (2)를 만족하는 모든 윈도우 $\vec{\chi}_{(m)j}$로 후보 집합을 구성해야 한다.

$$d\left(\vec{\chi}_{(m)j}, \vec{\tau}_{(m)j}\right) \le \frac{\epsilon}{\sqrt{p}} \tag{2}$$

질의 시에 주어진 이동평균 계수 m이 인덱싱 계수 k와 다른 경우, k-인덱스를 이용한 탐색 알고리즘을 설명하기 위하여 다음의 정리 1과 보조 정리 1이 필요하다.

정리 1 길이 n (≥ 3)인 임의의 시퀀스 $\vec{X} = (x_i)$, $\vec{T} = (t_i)$ $(0 \le i < n)$에 대하여 다음이 성립한다.

$$d\left(\vec{X}_{(m)}, \vec{T}_{(m)}\right) \ge d\left(\vec{X}_{(k)}, \vec{T}_{(k)}\right) \tag{3}$$

여기에서, $1 \le m \le k < n-1$이며, 다음과 같은 조건을 만족하여야 한다.

$$\forall i, x_i \ge t_i \ \vee \ \forall i, x_i \le t_i \ (0 \le i < n) \tag{4}$$

증명: 생략. □

보조 정리 1 길이 w (≥ 3)인 임의의 윈도우 $\vec{\chi}_j = (\chi_{ji})$, $\vec{\tau}_j = (\tau_{ji})$ $(0 \le i < w)$에 대하여 다음이 성립한다.

$$d\left(\vec{\chi}_{(m)j}, \vec{\tau}_{(m)j}\right) \le \frac{\epsilon}{\sqrt{p}} \Rightarrow d\left(\vec{\chi}_{(k)j}, \vec{\tau}_{(k)j}\right) \le \frac{\epsilon}{\sqrt{p}} \tag{5}$$

여기에서, $1 \le m \le k < w-1$이며, 다음과 같은 조건을 만족하여야 한다.

$$\forall i, \chi_{ji} \ge \tau_{ji} \ \vee \ \forall i, \chi_{ji} \le \tau_{ji} \ (0 \le i < n) \tag{6}$$

증명: 생략. □

이동평균 계수 m이 인덱싱 계수 k와 다른 경우, 공식 (5)의 결론부의 식인 공식 (1)을 만족하는 모든 윈도우 $\vec{\chi}_{(k)j}$로 후보 집합을 구성한다. 이때, 공식 (6)의 조건을 만족하여야 한다. 본 논문에서는 윈도우 $\vec{\chi}_{(k)j}$와 $\vec{\tau}_{(k)j}$를 각각 윈도우 $\vec{\chi}_{(m)j}$와 $\vec{\tau}_{(m)j}$의 매칭 윈도우*(matching window)*라 정의한다.

k-인덱스를 이용하여 공식 (1)을 만족하는 탐색을 수행할 때, 보조 정리 1을 통하여 착오 기각이 발생하지 않음을 보일 수 있다. 즉, 공식 (5)의 결론부의 조건을 만족하는 $\left(\vec{\chi}_{(k)j}, \vec{\tau}_{(k)j}\right)$ 쌍의 집합은 전제부의 조건을 만족하는 $\left(\vec{\chi}_{(m)j}, \vec{\tau}_{(m)j}\right)$ 쌍의 집합을 포함한다. 따라서, 결론부의 식에 의하여 영역 탐색을 수행할 때 착오 기각이 발생하지 않는다.

보조 정리 1이 성립하기 위한 조건인 공식 (6)이 만족되지 않는, 다음의 공식 (7)과 같은 경우에는 앞에서 설명한 영역 탐색 방법을 그대로 사용할 수 없다.

$$\exists i, l, \ \chi_{ji} > \tau_{ji} \ \wedge \ \chi_{jl} < \tau_{jl} \ (0 \le i, l < w, i \ne l) \tag{7}$$

이러한 경우에는 윈도우 $\vec{\chi}_j$ 또는 $\vec{\tau}_j$를 위/아래로 이동(shift)하여 공식 (6)이 만족되도록 한 다음, 이동한 거리에 따라 새로운 탐색 범위를 구하여 영역 탐색을 수행한다. 먼저, 다음과 같이 윈도우 $\vec{\chi}_j$와 $\vec{\tau}_j$의 최소값/최대값 간의 차이 값 d_1과 d_2를 정의한다.

$$d_1 = \min(\vec{\chi}_j) - \max(\vec{\tau}_j), \ d_2 = \min(\vec{\tau}_j) - \max(\vec{\chi}_j) \tag{8}$$

여기에서, $\min(\vec{\chi}_j)$와 $\max(\vec{\chi}_j)$는 윈도우 $\vec{\chi}_j$를 구성하는 값들 중 최소값과 최대값을 의미한다.

본 논문에서는 윈도우 $\vec{\chi}_j$를 이동하는 대상으로 한다. 윈도우 $\vec{\chi}_j$의 이동 거리 d_s는 다음의 공식 (9)와 같이 구한다.

$$d_s = \begin{cases} |d_1| & \text{if } |d_1| \le |d_2| \\ -|d_2| & \text{otherwise} \end{cases} \tag{9}$$

이동 거리 d_s가 정해지면 새로운 탐색 범위 ϵ'을 다음의 공식 (10)과 같이 구한다.

$$\epsilon' = \frac{\epsilon}{\sqrt{p}} + |d_s| \cdot \sqrt{w_m} \tag{10}$$

여기에서, w_m은 m-이동평균 변환된 윈도우 $\vec{\chi}_{(m)j}$, $\vec{\tau}_{(m)j}$의 길이이다.

공식 (6)의 조건을 만족하지 않는 경우, 다음의 공식 (11)을

만족하는 모든 윈도우 $\vec{\chi}_{(k)j}$로 후보 집합을 구성한다.

$$d\left(\vec{r}_{(k)j}, \vec{\chi}_{(k)j} + d_s\right) \leq \frac{\epsilon}{\sqrt{p}} + |d_s| \cdot \sqrt{w_m} \qquad (11)$$

다음의 보조 정리 2는 새로운 탐색 범위 ϵ'에 의한 영역 탐색이 착오 기각을 발생하지 않음을 보이기 위하여 필요하다.

보조 정리 2 윈도우 $\vec{r}_{(k)j}$, $\vec{\chi}_{(k)j}$가 $\vec{r}_{(m)j}$, $\vec{\chi}_{(m)j}$의 매칭 윈도우들이고, 윈도우 $\vec{\chi}_{(k)j} + d_s$가 윈도우 $\vec{\chi}_{(k)j}$를 양의 방향으로 d_s 만큼 이동한 윈도우라면 다음 식이 성립한다.

$$d\left(\vec{r}_{(m)j}, \vec{\chi}_{(m)j}\right) \leq \frac{\epsilon}{\sqrt{p}} \Rightarrow d\left(\vec{r}_{(k)j}, \vec{\chi}_{(k)j} + d_s\right) \leq \epsilon' \qquad (12)$$

여기에서, 이동 거리 d_s는 공식 (9)에 의하여 구해진 값이며, w_m은 윈도우 $\vec{r}_{(m)j}$, $\vec{\chi}_{(m)j}$의 길이이다.
증명: 생략. □

공식 (12)의 결론부의 조건을 만족하는 $\left(\vec{\chi}_{(k)j}, \vec{r}_{(k)j}\right)$ 쌍의 집합은 전제부의 조건을 만족하는 $\left(\vec{\chi}_{(m)j}, \vec{r}_{(m)j}\right)$ 쌍의 집합을 포함한다. 따라서, 결론부의 식에 의하여 영역 탐색을 수행할 때 착오 기각이 발생하지 않는다.

4 성능 평가

본 실험의 목적은 하나의 k-인덱스만을 이용하더라도 모든 이동평균 계수 m에 대하여 인덱스가 생성되어 있는 경우에 비해 제안된 알고리즘의 탐색 성능이 크게 저하되지 않음을 보이기 위함이다.

본 실험에 사용한 데이타베이스는 1994년 11월부터 1998년 5월까지 길이 1024의 한국 주가 데이타 620 개 종목의 데이타 시퀀스로 구성된다. 질의 시퀀스는 임의의 128 개 종목의 데이타 시퀀스들로부터 임의 위치에서 길이 256의 서브시퀀스 추출하고 변형하여 생성하였다. 탐색 범위 ϵ은 탐색 결과 선택률(selectivity)을 기준으로 결정하였으며, 실험에 사용된 선택률 값들은 0.0001, 0.001, 0.01, 0.1이다.

제안된 알고리즘의 실험을 위하여 이동평균 계수 $k = 128$에 대하여 k-인덱스를 생성하고, 참고문헌 [4]의 알고리즘을 확장없이 적용하는 경우의 실험을 위하여 질의 시에 주어지는 이동평균 계수 $m = 8i$ $(i = 1, 2, \ldots, 16)$인 경우와 $m = 1, k - 1$인 경우에 대하여 일반 인덱스를 생성하였다. 인덱싱을 위하여 $f = 5$의 인덱스 차원을 결정하였다.

첫번째 실험은 m-이동평균 변환 서브시퀀스 매칭 질의가 주어졌을 때, 이동평균 계수 m에 대한 인덱스가 생성되어 있지 않은 경우와 생성되어 있는 경우에 탐색 알고리즘의 실행 시간(elapsed wall-clock time)을 비교하는 실험이다. 그림 1은 첫번째 실험에 의해 얻어진 결과를 보인 것이다. 가로 축은 질의 시의 이동평균 계수 m을 나타내고, 세로 축은 인덱스가 없는 경우의 실행 시간 t_k를 인덱스가 있는 경우의 실행 시간 t_m으로 나눈 값을 나타낸다. 그림 1에 보인 실행 시간 비율 값은 128 개의 질의 시퀀스에 대하여 얻어진 실행 시간 비율 값들을 평균한 값이다.

두번째 실험은 m-이동평균 변환 서브시퀀스 매칭 질의가 주어졌을 때, 제안된 알고리즘과 순차 검색 알고리즘의 실행 시간을 비교하는 실험이다. 그림 2는 두번째 실험에 의해 얻어진 결과를 보인 것이다. 가로 축은 이동평균 계수 m을 나타내고, 세로 축은 제안된 알고리즘이 k-인덱스를 이용하여 탐색에 걸린 실행 시간 t_k를 순차 검색 알고리즘의 실행 시간 t_{seq}로 나눈 값을 나타낸다.

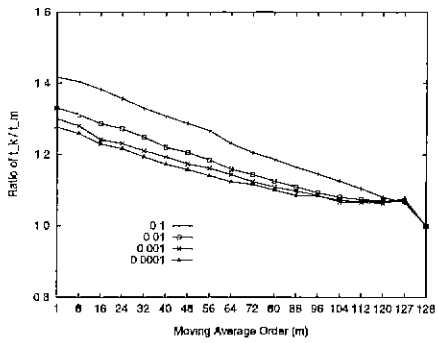

그림 1: 제안된 알고리즘의 실행 시간 비교.

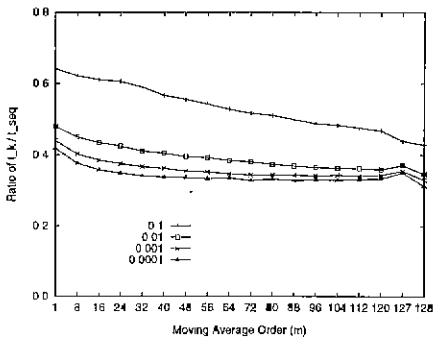

그림 2: 순차 검색 알고리즘과의 실행 시간 비교.

참고문헌

[1] Agrawal, R. et al., "Efficient Similarity Search in Sequence Databases," In *Proc. Foundations of Data Organization and Algorithms Conference*, pp. 69-84, Chicago, Illinois, Oct. 1993.

[2] Agrawal, R. et al., "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*, pp. 490-501, Zurich, Switzerland, Sept. 1995.

[3] Chatfield, C., *The Analysis of Time Series: An Introduction*, 3rd Ed., Chapman and Hall, 1984.

[4] Faloutsos, C. et al., "Fast Subsequence Matching in Time-Series Databases," In *Proc. Int'l Conf. on Management of Data*, ACM SIGMOD, pp. 419-429, Minneapolis, Minnesota, June 1994.

[5] Kendall, M., *Time-Series*, 2nd Ed., Charles Griffin and Company, 1976.

[6] Rafiei, D. and Mendelzon, A., "Similarity-Based Queries for Time Series Data," In *Proc. Int'l Conf. on Management of Data*, ACM SIGMOD, pp. 13-25, Tucson, Arizona, June 1997.