맵리듀스를 이용한 SE Matching 기반 유사 검색 기법

권정민, 서준일, 안진우, 이정준
한국산업기술대학교 컴퓨터공학과
e-mail:cockrobinest@gmail.com, finalnight@naver.com,
ajw21c@naver.com, jjlee@kpu.ac.kr

Similar Sequence Matching based on SE Matching with MapReduce

Jeong-Min Kwon, Jun-Il Seo, Jin-Woo An, Jeong-Joon Lee
Dept of Computer Engineering, Korea Polytechnic University

요 약

시계열 데이터 검색은 금융, 생명정보 등 많은 분야에서 요구되는 주요한 기술로서 연구되어 왔다. 특히 기존에 제안된 스케일링(Scaling)과 쉬프팅(Shifting)을 이용한 검색인 SE Match는 유사 서브시퀀스를 효과적으로 찾아내는 방법으로 알려지고 있다. 본 논문에서는 이 방법에 맵리듀스를 적용하는 MRSE-검색 (MapReduce-based Searching with Shift-Eliminated)방법을 제안한다. 본 논문이 제안하는 방법으로 분산처리를 통하여 응답시간의 개선과 대용량의 시계열 데이터에서 효율적인 검색이 가능 할 것으로 사료된다.

1. 서론

시계열 데이터란 시간이 경과함에 따라 일정 시간 간격으로 측정되는 실수 값 데이터를 의미한다. 시계열 데이터의 종류로는 주가 지수 데이터, 온도 변화 데이터, 경제 성장률 데이터, 지진 데이터 등이 있으며 이를 분석하는 방법으로는 유사 시퀀스 매칭[1,2,5,7]이 있다. 유사 시퀀스 매칭은 시퀀스를 서브 시퀀스와 질의 시퀀스로 나누어 유사한 시퀀스를 탐색하는 기법이다.

유사 시퀀스 매칭은 시퀀스를 서브 시퀀스와 질의 시퀀스로 나누어 유사한 시퀀스를 탐색하는 기법으로 계산 요구량이 매우 크다. 그런데 시간이 경과함에 따라 시계열 데이터는 대용량화하고, 금융 등의 분야에서는 실시간 검색결과를 요구하고 있다. 이러한 환경변화와 요구사항에 비하여 기존의 유사 시퀀스 매칭은 충분히 대응하고 있지 못하다.

맵리듀스는 분산 컴퓨팅을 지원하기 위한 목적으로 개발된 프레임워크로 페타바이트 이상의 빅데이터를 신뢰할 수 없는 컴퓨터로 구성된 클러스터 환경에서 병렬 처리를 지원한다.

본 논문에서는 시계열 데이터의 시퀀스 매칭을 위한 유사 시퀀스 매칭 모델 중 SE Match 모델[2,3]을 정의하고, 이 모델을 빠른 시간에 처리할 수 있도록 맵리듀스 프레임워크인 하둡을 이용한 방식에 대해서 논의한다. 또한 제안된 방안의 우수성을 규명하기 위해 다양한 실험을 통해 매칭 결과와 성능을 제시한다.

2. 관련연구

2.1 유사 시퀀스 매칭

유사 시퀀스 매칭은 질의 시퀀스를 받아 데이터 시퀀스를 질의 시퀀스와 동일한 길이로 슬라이딩 윈도우를 구성하여 질의 시퀀스와 구성된 슬라이딩 윈도우를 비교하여 유사한 시퀀스를 검색하는 기법이다.[1,2,5,7]

전체 데이터 시퀀스를 $\vec{S}$, 서브 시퀀스를 $\vec{P}$, 질의 시퀀스를 $\vec{Q}$, 데이터 시퀀스의 길이를 n, 슬라이딩 윈도우의 길이를 w, i번째 서브시퀀스를 $\vec{P_i}$ 이라 구성한다. <표 1>은 본 논문에서 사용되는 표기법을 정리한 것이다.

<표 1> 표기법 정리

기 호	정 의
$\vec{S}$	전체 데이터 시퀀스
$\vec{P}$	서브 시퀀스
$\vec{Q}$	질의 시퀀스
$\vec{N}$	단위 벡터 , 전체 엔트리가 1인 시퀀스
$\vec{P_i}$	i번째 서브시퀀스
n	서브 시퀀스의 길이
w	슬라이딩 윈도우의 길이

2.2 순차검색

순차검색[1, 2, 4]은 길이가 $1 \leq n$인 서브시퀀스 $\overrightarrow{P}=\{p_1,p_2,p_3 \cdots p_n\}$ 와 질의시퀀스 $\overrightarrow{Q}=\{q_1,q_2,q_3, \cdots q_n\}$를 차원 공간상의$n$ 한 점으로 표현하여 (식 1)을 사용하여 두 시퀀스 간의 유클리디언 거리 D(Euclidean Distance)[1,2,3,5]를 계산한다.

$$D=\sqrt{\sum_{i=1}^{n}(p_i-q_i)^2} \qquad \text{(식 1)}$$

전체 데이터 시퀀스 $\overrightarrow{S}$를 질의 시퀀스와 동일한 길이의 서브 시퀀스 $\overrightarrow{P}$의 슬라이딩 윈도우를 생성하여 생성된 모든 서브 시퀀스와 질의 시퀀스에 대하여 유클리디언 거리를 계산한다. 비교하는 두 시퀀스간의 유클리디언 거리가 사용자가 입력한 유사 허용치보다 작다면 두 시퀀스는 유사하다고 판정한다.

2.3 SE Match

SE Match는 시퀀스를 구성하는 모든 엔트리에 동일한 실수 값을 곱하여 시퀀스의 진폭에 변화를 주는 스케일링(Scalling)과 시퀀스를 구성하는 모든 엔트리에 동일한 실수 값을 증감하여 시퀀스를 수직으로 이동시키는 쉬프팅(Shifting)을 지원하는 매칭기법이다.[2,3]

먼저 질의 시퀀스와 서브 시퀀스 모두 위상의 개념을 없애기 위해 (식 2)를 사용하여 시퀀스의 각 엔트리 값을 시퀀스 전체 엔트리의 평균값만큼 감소시킨다.

$$T_{se}(\overrightarrow{P})=\overrightarrow{P}-\frac{\overrightarrow{P}\bullet\overrightarrow{N}}{\|\overrightarrow{N}\|^2}\overrightarrow{N} \qquad \text{(식 2)}$$

그리고 (식 3)을 사용하여 두 벡터의 크기를 동일하게 만들어 줄 수 있는 Scaling Factor a 값을 구한다.

$$a=\frac{T_{se}(\overrightarrow{P})\bullet T_{se}(\overrightarrow{Q})}{\|T_{se}(\overrightarrow{P})\|^2} \qquad \text{(식 3)}$$

그 후 (식 4)를 사용하여 시퀀스 $\overrightarrow{Q}$와 스케일링(Scaling) 된 시퀀스 $\overrightarrow{P}$의 각 엔트리의 차의 평균 값인 Shifting offset b 를 구한다.

$$b=\frac{(\overrightarrow{Q}-a\bullet\overrightarrow{P})\bullet\overrightarrow{N}}{\|\overrightarrow{N}\|^2} \qquad \text{(식 4)}$$

이후 $(a\bullet\overrightarrow{P}+b)$와 $\overrightarrow{Q}$의 유클리디언 거리 D를 구하여 계산된 거리가 사용자가 입력한 유사 허용치보다 작거나 같으면, 두 시퀀스는 유사하다고 판정한다.

2.4 맵리듀스

맵리듀스[6]는 맵함수과 리듀스함수로 나누어 두 함수의 조합을 이용해 분산/병렬 시스템을 지원한다.

맵함수는 위치를 나타내는 오프셋(Offset)을 키의 입력으로, 오프셋(Offset)내의 데이터를 값으로 전달 받는다. 맵의 입력으로 키(K1), 값(V1)이 전달되면 맵은 전달된 키-값 페어를 이용하여 사용자의 맵함수 로직을 처리한 후 새로운 키(K2), 값의 목록(LIST(V2))을 출력한다.

리듀스는 맵으로부터 출력된 키(K2), 값의 목록(K2, LIST(V2))를 입력으로 전달받고 사용자의 리듀스함수 로직을 처리한 후 최종으로 값의 목록(LIST(V3))을 출력한다.

<표 2> 맵-리듀스 입출력 키-값

map(K1, V1) → K2, LIST(V2)
reduce(K2, LIST(V2)) → LIST(V3)

3. 맵리듀스를 응용한 SE Match 기반 유사 검색 기법

3.1 개념

본 논문에서는 맵리듀스를 응용한 SE Match 기반 유사 검색 기법에 대해 논의하며 이를 MRSE-검색(MapReduce-based Searching with Shift-Eliminated)이라 명명한다.

유사 시퀀스 매칭의 기본 구조는 사용자로부터 질의 시퀀스를 입력받아 유사 시퀀스 매칭 프로그램을 실행하여 사용자가 입력한 유사 허용치이내인 데이터만 추출한다.

본 논문에서 맵과 리듀스의 작업을 제어하는 Job과 유사 시퀀스 매칭을 수행하는 맵들과 추출된 데이터를 모으는 리듀스들로 구성하여 작업을 수행한다.

Job은 사용자로부터 질의 시퀀스를 입력받아 맵들로 전송한다. 맵들은 질의 시퀀스를 기준으로 HDFS에 저장되어 있는 데이터로부터 유사 시퀀스 매칭을 수행하고 유사 허용치 이내인 데이터를 리듀스들로 전달한다. 리듀스들은 데이터를 키를 기준으로 정렬하여 최종으로 데이터를 출력한다.

본 논문에서 맵의 입력키는 데이터의 오프셋(Offset)번호로 하였으며 입력값은 한 라인의 텍스트로 하였다. 맵의 출력키는 서브 시퀀스 데이터의 시작연도로 하였으며 출력값은 SE Match를 통해 계산된 유클리디언 거리 D, 시작날짜와 시간으로 하였다.

리듀스의 입력값은 맵의 출력값과 동일하며 리듀스의 출력값은 리듀스의 입력값 중 유클리디언 거리 D를 기준으로 오름차순으로 정렬하여 연도별 상위 5개만 출력하도록 하였다. 다음 <표 2>는 맵과 리듀스의 입출력 키-값 페어를 정리한 것이다.

<표 2> 맵과 리듀스의 입출력 키-값 페어

```
map (K1, V1, K2, LIST(V2)){
    K1 ← data offset number
    V1 ← 1 Line Text
    K2 ← Sub Sequence Start Year
    V2 ← D, Sub Sequence Start Date, Time
}
reduce (K2, LIST(V2), K2, LIST(V3)){
    K2 ← Sub Sequence Start Year
    V2 ← D, Sub Sequence Start Date, Time
    K2 ← Sub Sequence Start Year
    V3 ← D, Sub Sequence Start Date, Time
}
```

3.2 알고리즘

본 논문에서는 유사 시퀀스 매칭 기법으로 SE Match를 사용한다. SE Match에서 중요한 부분은 세 가지로, 위상을 제거하는 부분과 Scaling Factor a를 구하는 부분, Shifting Offset b를 구하는 부분이다.

위상을 제거한다는 것은 시퀀스 엔트리를 매개변수로 전달 받아 평균값을 구한 후 시퀀스의 모든 엔트리를 평균값만큼 감소시키는 것으로 의사코드는 <표 3>과 같다.

<표 3> 위상제거 의사코드

```
shifteliminate(int[] sequence)

 int i ← 0;
   double ave ← CALCULATE the average of array
of sequence;

 while i < sequence.length do
 begin
   (double)sequence[i] ← sequence[i]-ave;
 end;

 return sequence;
```

Scaling Factor a를 구한다는 것은 위상을 제거한 서브 시퀀스 엔트리와 질의 시퀀스 엔트리를 매개변수로 전달 받은 후 두 엔트리를 곱하여 누적한 값을 질의 시퀀스 엔트리를 제곱하여 누적한 값으로 나누는 것으로 의사코드는 <표 4>와 같다.

<표 4> Scaling Factor a 구하는 의사코드

```
scailing_qs_a(double[] Tse_qs, double[] Tse_ss)

 int i ← 0;
 double Tse_qs_ss_multiplied[];
 double Tse_qs_squared[];

 while i < Tse_qs.length do
 begin
  Tse_qs_ss_multiplied ←
       Tse_qs_ss_multiplied + Tse_qs[i]*Tse_ss[i];
  Tse_qs_squared ←
       Tse_qs_squared + Tse_qs[i]*Tse_qs[i];
 end;

 return Tse_qs_ssmultiplied/Tse_qs_squared;
```

Shfting Offset b를 구한다는 것은 기존 서브 시퀀스의 모든 엔트리에서 기존 질의 시퀀스의 모든 엔트리에 Scaling Factor a를 곱하여 변경된 질의 시퀀스 엔트리 값을 감소하여 감소된 값을 누적한 후 질의 시퀀스 길이로 나누는 것으로 의사코드는 <표 5>와 같다.

<표 5> Shifting Offset b 구하는 의사코드

```
shifting_offset_qs_b(int qs[],int ss[])

 int i ← 0;
 double offset_b;

 while i < qs.length do
 begin
    offset_b ← offset_b + (ss[i] - (qs_a*qs[i]));
 end;

 return offset_b/qs.length;
```

4. 결론

본 논문에서는 유사 시퀀스 매칭을 위해 제안된 스케일링과 쉬프팅을 이용한 매칭 기법인 SS-검색기법에 하둡 맵리듀스를 적용하는 방법을 제안한다. 제안하는 방법은 기존 방법에서 슬라이딩 윈도우를 구성하여 순차적으로 검색하는 방법이 갖는 많은 계산량을 맵리듀스 기법으로 분산 처리하는 방법이다.

본 논문에서 제안하는 방법은 분산처리를 통하여 응답시간 개선 효과와 대용량 데이터에도 SS-검색 기법을 적용할 수 있는 실제 적용 가능한 응답시간을 줄 것으로 예상한다. 현재 MRSE-검색 방법의 효율성 검증을 위해 실험모델에 따른 실험을 진행 중이며, 향후 실제 타임시리즈

데이터등을 이용한 실험을 진행할 예정이다.

참고문헌

[1] D. Rafiei and A. Mendelzon, "Similarity-Based Queries for Time-Series Data," In Proc. Int'l. Conf. on Management of Data, ACM SIGMOD, pp. 13-24, 1997.

[2] 유승근, 이상호, "한국 주식 데이터를 이용한 서브시퀀스 매칭 방법의 효과성 평가", 정보처리학회논문지D, 제12-D권 제3호, pp. 355-364, 2005.

[3] Kelvin Kam Wing Chu and Man Hon Wong, "Fast Time Series Searching with Scaling and Shifting," In Proc. Int'l. Symp. on Principles of Database Systems, ACM PODS, pp. 237-248, 1999.

[4] 문양세, 김진호, "시계열 데이터베이스에서 단일 색인을 사용한 정규화 변환 지원 서브시퀀스 매칭", 한국정보과학회 2005 한국컴퓨터종합학술대회 논문집(B), pp. 157-159, 2005.

[5] Christos Faloutsos, M. Ranganathan and Yannis Manolopoulos, "Fast Subsequence Matching in Time-series Database," In Proc. Int'l. Conf. on Management of Data, ACM SIGMOD, pp. 419-429, 1994.

[6] 곽재혁, 윤준원, 정용환, 함재균, 박동인 "하둡 기반 천문 응용 분야 대규모 데이터 분석 기법", 정보과학회 논문지 : 컴퓨팅의 실제 및 레터, 제17권 제11호, pp.587-591, 2011.

[7] Rakesh Agrawal and Christos Faloutsos, Arun Swami "Efficient Similarity Search In Sequence Databases," In Proc. Int'l. Conference on Foundations of Data Organization and Algorithms, FODO, pp. 69-84, 1993.