데이터 품질 관리 : CRM을 사례로 연산자와 매칭기법 중심

Data Quality Management: Operators and a Matching Algorithm with a CRM Example

심준호(Junho Shim)*

초 록

CRM 과 같은 전자상거래응용시스템에서 동일한 데이터의 중복이나 불일치는 종종 일어나며 이는 바람직하지 못하다. 데이터 품질 관리란 데이터들간의 비일치와 중복을 발견하고 제거함을 목적으로 한다. 통상적인 데이터 품질관리 프로세스는 클리닝, 매칭, 통합의 세 단계를 거친다. 본 논문에서는 일반적인 데이터 품질 관리를 각 단계별로 필요한 연산자들을 정의한다. 특히 실제적인 시스템 구현에서 필요한 매칭 단계에서 사용하는 거리함수와 매칭 알고리즘을 제안하며, 마지막으로 관련 연구를 제시한다.

ABSTRACT

It is not unusual to observe that there is a great amount of redundant or inconsistent data even within an e-business system such as CRM(Customer Relationship Management) system. This problem becomes aggravate when we construct a system of which information are gathered from different sources. Data quality management is indeed needed to avoid any possible redundant or inconsistent data in such information system. A data quality process, in general, consists of three phases: data cleaning (scrubbing), matching, and integration phase. In this paper, we introduce and categorize data quality operators for each phase. Then, we describe our distance function used in the matching phase, and present a matching algorithm PRIMAL (a PRactical Matching Algorithm). And finally, we present a related work and future research.

키워드 : 데이터 품질관리, 데이터 클리닝, 고객관리시스템
Data Quality, Data Cleaning, CRM

---

* 본 연구는 숙명여자대학교 2002년도 교내연구비 지원에 의해 수행되었음.
* 숙명여자대학교 정보과학부 조교수

1. 서 론

전자상거래시스템이나 데이터웨어하우싱과 같은 데이터 통합 시스템에서 최근 관심을 받는 분야 중 하나는 데이터 품질 관리(data quality management) 이다 [11,10]. 예를 들어 전자상거래 응용기술의 가정 대중적인 예라 할 수 있는 CRM(Customer Relationship Management) 시스템을 보면, 시스템 데이터웨어하우스로 로딩되는 데이터의 품질의 높고 낮음이 시스템의 성공적인 운영을 위한 기초가 된다. [7,10] 통합되어지는 데이터들은 서로 다른(heterogeneous) 데이터베이스로부터 오기도 하고 동일한(homogeneous) 데이터베이스 내에서도 이루어 지도 하는데, 문제는 통합되어지는 데이터에는 상당 부분 비일치(혹은 비일관, inconsistent)하고 중복(redundant)된 정보가 존재하는데 있다. 데이터 품질 관리시스템의 한 목표는 이러한 비일치하고 중복된 데이터를 발견하고 처리하는 것이다.

비일치하고 중복된 데이터를 처리를 위한 데이터 품질 관리는 통상적으로 *데이터 클리닝(cleaning), 매칭(matching), 통합(integration)*의 세 단계로 이루어 진다 [9,8] *데이터 클리닝* (혹은 스크러빙(scrubbing) 이라 불림)이란 데이터 도메인과 데이터 값을 참조하여 데이터 포맷 표준화, 에러 교정 등을 하는 단계이다. 클린된 데이터는 일치하는지 중복되었는지를 목적으로 다른 데이터와 그 값이 *매칭*되게 된다. 매칭된 결과로부터 동일하거나 비슷한 데이터들은 일치되고 중복을 제거하기 위해 클래스화(classification) 혹은 클러스터링(clustering) 되어 하나의 데이터로 통합(*integrated*) 된다.

예를 들어 CRM 고객정보 데이터베이스에 존재하는 인물과 학력에 대해 데이터의 일관성과 중복에 대한 품질 관리를 살펴보자. 〈그림 1〉

인물

주민등록 번호	이름	성별	...
501018-1400000	홍순자	여	...
600101-2300000	김미례	여	...
590201-1259800	박미자	여	...

학력

주민등록 번호	입학 년도	졸업 년도	출신학교 및 전공
...	...	...	...
501018-1400000	1965	1968	숙명여자고등학교
501018-1400000	1968	1972	숙명여자대학교 약학과 학사
...	...	...	...
600101-2300000	78	82	숙명여대 경영학과 학사
...	...	...	...
590201-1259800	1982	1984	숙명여자대학 법학과 석사
...	...	...	...

〈그림 1〉 중복되고 비일관된 CRM 고객정보 데이터베이스의 예

첫번째 단계인 데이터 *클리닝단계*에서는 각 테이블의 컬럼 도메인, 포맷 표준, 가능한 에러 교정을 한다. 학력 테이블의 입학년도와 졸업년도가 도메인이 DATE나 숫자로서 년도만 가진다고 하면 주민등록번호가 590201-1259800 인 레코드는 해당 필드값이 도메인 규약(domain constraint)을 위반하므로 1982와 1984로 바꾸어 준다. 주민등록번호가 600101-2300000 인 레코드는 78과 82를 각각 1978과 1982로 바꾸어 준다. 출신학교 및 전공 필드는 한글이외에 쉼표나 괄호 등을 갖지 않도록 포맷을 표준화 시켜준다. 또 인물 테이블에서 590201-1259800은 주민등록번호나 성별 등의 도메인 값은 유효하나 주민등록번호 뒤 8자리가 1로 시작하면 남자를 뜻하므로 둘 중의 한 값은 오류일 것이다. 데이터 클리닝 과정에서 사용자로부터의 확인을 통해 이와 같은 데이터 에러도 수정되어 질 수 있다. 클리닝 과정에서는 필요한 스키마에 대한 변경도 이루어질 수 있다. 예를 들어 출신학교 및 전공은 데이터를 분석해 학교명, 학교분류, 전공, 학위 등의 필드로 나뉘어 질 수 있다.

고객정보 데이터베이스에 출신학교별로 인물을 분류할 수 있도록 하려면 동일한 출신학교에 대한 여러 상이한 데이터 값을 하나의 대표 값으로 균일하게 나타낼 수 있어야 한다.

학력

주민등록번호	입학년도	졸업년도	학교명	학교분류	전공	학위
...	...	...	...	...	...	...
600101-2300000	1978	1982	숙명여대	대학교	경영학과	학사
501018-2400000	1965	1968	숙명여자고등학교	고등학교		
590201-2259800	1982	1984	숙명여자대학	대학교	법학과	석사
501018-2400000	1968	1972	숙명여자대학교	대학교	약학과	학사
...	...	...	...	...	...	...

주민등록번호	입학년도	졸업년도	학교명	학교분류	전공	학위
...	...	...	...	...	...	...
600101-2300000	1978	1982	숙명여자대학교	대학교	경영학과	학사
501018-2400000	1965	1968	숙명여자고등학교	고등학교		
590201-2259800	1982	1984	숙명여자대학교	대학교	법학과	석사
501018-2400000	1968	1972	숙명여자대학교	대학교	약학과	학사
...	...	...	...	...	...	...

〈그림 2〉 클리닝, 매칭, 통합과정을 거친 학력 테이블의 예

예를 들어 숙명여자대학교, 숙명여대, 숙명여자대학 등은 모두 동일한 학교를 지칭하는 것으로서 분류될 수 있어야 한다. 문제는 이러한 작업을 전체 탐색과 수작업에 의하지 않고 시스템이 일정 부분 해줄 수 있도록 한다는데 있다. *매칭과정*에서는 학교명이 얼마나 서로 비슷한지를 보는데, 레코드를 학교명 필드에 의해 정렬한 후 부근에 있는 레코드 들의 학교명 필드를 적당한 매칭함수에 의해 4개의 학교명이 비슷하다는 것을 알아낸다.

마지막 과정인 *통합과정*에서는 이 4개의 학교명이 하나의 분류나 클러스터로 될 수 있는지를 결정하고 그 클러스터의 대표 값으로 모든 데이터 필드를 균일하게 치환하게 된다. 예를 들어 숙명여자고등학교와 다른 세개 학교명 값과는 설사 매칭함수는 높은 유사성을 보였더라도 학교분류가 다르다는 지식에 기반하여 다른 세 개와는 다른 클러스터에 속하게 된다.

본 논문의 공헌

데이터 품질 관리 연구 영역에 있어 본 논문이 공헌하는 점들은 다음과 같다.

첫째는 데이터 품질 관리 과정에서 데이터의 일관성과 중복 처리를 위하여 각 단계별로 필요한 연산자(operator)를 분석, 정리 한다. 제시된 연산자는 [9,3]등의 연구에서 제시된 연산자들을 분석하고, 실제 데이터 품질 관리를 국내 사례에 적용한 경험을 토대로 미비하거나 혹은 중요도 등을 고려하여 재 구성한데 그 의의가 있다. 각 연산자는 CRM 고객정보 테이블을 중심으로 일일이 예를 통해 설명함으로서 그 이해도를 높인다.

둘째는 매칭과정에서 사용된 PRIMAL(a PRactical Matching ALgorithm) 이라는 매칭 알고리즘의 제안에 있다. PRIMAL 은 한글로 구성된 학교명등과 같은 필드 값을 효율적이고 정확하게 매칭하도록 제안된 알고리즘으로서, 기존의 데이터 품질 시스템연구에서와 같이 라틴 언어 중심의 edit-distance 함수와 distance-filtering 기법에 근거를 둔 방식과 그 차별이 있다.

이를 위한 본 논문의 구조는 다음과 같다. 2장에서는 데이터 품질 관리 시스템에서 필요한 데이터 품질 관리 연산자에 대해 논한다. 3장에서는 PRIMAL 이라는 새로운 매칭 알고리즘을 소개한다. 4장에서는 데이터 품질 관리와 CRM 도메인에서의 관련 연구를 서베이(survey) 형식으로 소개한다. 마지막으로 5장은 결론 및 장래 연구 방향에 대해 기술한다.

2. 데이터 품질 관리 연산자

데이터 품질 관리를 위해 필요한 연산자는 품질 관리 과정이 세 과정으로 나뉜 것과 같이 크게 1) 데이터 클리닝 연산자 2) 데이터 매칭 연산자 3) 데이터 통합 연산자 의 세가지로 나누어 살펴 볼 수 있다. 주의해야 할 점은 여기에서 제시되는 연산자들은 논리적인 연산자 (logical operator)이며, 이들 연산자가 어떻게 구현되어야 하는지를 나타내는 물리적 연산자(physical operator)는 아니다는 점이다. 실제 데이터 품질 관리 시스템에서는 아래에

서 제시되는 논리적인 연산자가 실제로 구현된 물리적인 연산자에 매핑되어져야 한다.

2.1.클리닝 연산자

클리닝 연산자란 데이터 클리닝 과정에서 필요한 스키마의 변환, 데이터 포맷 표준화, 데이터 분류, 데이터 에러 값 변환 등을 위해 필요한 연산자이다. 본 논문에서는 각 연산자를 정의하고, 필요하다고 생각되는 연산자에는 예를 달아 그 이해를 높이도록 한다.

1. 필드 추가 (add) 연산자

정의: $\alpha(R,x)=\{(a_1,\ldots,a_n,x)\mid(a_1,\ldots,a_n)\in R\}$

테이블 R에 새로운 필드 x를 추가하는 것이다. 각 레코드의 새 필드 x의 값은 NULL이나 시스템 디폴트 값으로 채워지게 된다.

2. 필드 제거 (drop) 연산자

정의: $\pi(R,i)=\{(a_1,\ldots,a_{i-1},a_{i+1},\ldots,a_n)\mid (a_1,\ldots,a_n)\in R\}$

테이블 R에 존재하는 i번째 필드 a_i를 제거한다.

3. 필드 복사 (copy) 연산자

정의: $\kappa(R,i,j)=\{(a_1,\ldots,a_i,\ldots,a_{j-1},a_i,a_{j+1},\ldots,a_{n+1})\mid(a_1,\ldots,a_n)\in R\}$

테이블 R에 존재하는 i번째 필드 a_i를 j번째 필드에 그대로 복사한다.

4. 필드 분리 (split) 연산자

정의: $\omega(R,i,spliter)=\{(a_1,\ldots,a_{i-1},left(a_i,spliter),right(a_i,spliter),a_{i+1},\ldots,a_n)\mid (a_1,\ldots,a_n)\in R\}$

테이블 R에 존재하는 i번째 필드 a_i를 스트링 위치 정수값이나 레귤라 수식인 spliter로 좌측 left(a_i,spliter)과 우측 right(ai,spliter)으로 분리한다.

예를 들어 그림 3은 학력 테이블의 4번째 필드인 학교명에 spliter 가 ' ' 인 경우의 ω(학력, 4, ' ') 연산을 적용한 결과를 보여준다.

5. 필드 나누기 (divide) 연산자

정의: $\delta(R,i,pred)=\{(a_1,\ldots,a_{i-1},a_{i+1},\ldots,a_n,a_i,null)\mid(a_1,\ldots,a_n)\in R\wedge pred(a_i)\}\cup\{(a_1,\ldots,a_{i-1},a_{i+1},\ldots,a_n,null,a_i)\mid(a_1,\ldots,a_n)\in R\wedge\neg pred(a_i)\}$

테이블 R에 존재하는 i번째 필드 a_i의 값이 프레디킷 pred를 만족하는지에 따라 만족하는 레코드와 만족하지 않는 레코드를 별도로

숙명여자고등학교
숙명여자대학교 약학과 학사
숙명여대 경영학과 학사
숙명여자대학 법학과 석사

숙명여자고등학교		
숙명여자대학교	약학과	학사
숙명여대	경영학과	학사
숙명여자대학	법학과	석사

〈그림 3〉 필드 분리 연산자 ω의 적용 예

각각 하나의 필드로 구분해 만든다.

예를 들어 그림 4는 학교명 필드에 pred가 '고등학교' 란 값을 가지고 있는지 아닌지를 보아 만족하는 것과 만족하지 않은 것을 따로 구분해 만든 것이다.

6. 필드 머지 (merge) 연산자

정의: $\mu(R,i,j)=\{(a_1,...,a_{i-1},a_{i+1},...,a_{j-1},a_{j+1},...,a_n,a_i\oplus a_j)\mid(a_1,...,a_n)\in R\}$

i번째 필드 a_i와 j번째 필드 a_j를 하나의 필드로 합한다.

7. 필드 포맷 (format) 연산자

정의: $\phi(R,i,f)=\{(a_1,...,a_{i-1},a_{i+1},a_n,f(a_i))\mid(a_1,...,a_n)\in R\}$

i번째 필드 a_i에 포맷 함수 f를 적용시킨다.

8. 검색 (select) 연산자

정의: $\sigma(R,pred)=\{(a_1,...,a_n)\mid(a_1,...,a_n)\in R\wedge pred(a_1,...,a_n)\}$

테이블 R 의 레코드들에서 프레디킷 pred를 만족하는 레코드만을 검색한다.

9. 폴딩/언폴딩 (fold/unfold) 연산자

정의: $\lambda(R,i_1,i_2,...,i_k)=\{(a_1,...,a_{i1-1},a_{i1+1},...,a_{i2-1},a_{i2+1},...,a_{ik-1},a_{ik+1},...a_n,a_{ij})\mid(a_1,...,a_n)\in R\wedge 1\le l\le k\}$

테이블 R의 필드 $i_1,i_2,\cdots,i_k$를 하나의 필드 i_j로 폴딩 하는것이다. 이 연산자는 멀티데이타베이스 분야에서 여러 스키마를 통합하기 위해 SchemaSQL이 제안한 연산자로서, 여러 필드를 하나의 필드로 만드는 것이다. 언폴드(unfold)는 반대로 하나의 필드를 여러 필드로 나누어 놓는 것이다. 그림 5는 언폴드의 예로 2,3번째 필드의 값에 따라, 각 필드의 값이 독자적인 필드로 되도록 레코드를 수평으로 펴놓은 것이다.

10. 테이블 맵핑(mapping) 연산자 : M

정의: $\eta(R,i_1,i_2,...,i_k)=\{(a_{i1},a_{i2},...,a_{ik})\mid(a_1,...,a_n)\in R\wedge 1\le k\le n\}$

테이블 R의 $i_1,i_2,\cdots,i_k$ 필드만을 뽑아 다른 테이블로 매핑(mapping) 하는 것이다. SQL문의 VIEW 연산자라고 생각하면 된다.

이상으로 데이터 품질 관리를 위한 10가지 클리닝 연산자들을 살펴보았다. 유의할 점은 이들 연산자들은 SQL을 지원하는 데이터베이스에 따라 그 물리적인 구현이 다른 방법으로 가능하다는 것이다. 예를 들어 Oracle 데이

숙명여자고등학교		
숙명여자대학교	약학과	학사
숙명여대	경영학과	학사
숙명여자대학	법학과	석사

숙명여자고등학교			
	숙명여자대학교	약학과	학사
	숙명여대	경영학과	학사
	숙명여자대학	법학과	석사

〈그림 4〉 필드 나누기 연산자 δ의 적용 예

주민등록 번호		
501018-2400000	1968	고등학교
590201-2259800	1984	대학교
501018-2400000	1972	대학교
501018-2400000	1965	중학교

언폴딩 →

← 폴딩

주민등록 번호	중학교	고등학교	고등학교
501018-2400000	1965	1968	1972
590201-2259800			1984

〈그림 5〉 폴딩과 언폴딩 연산자 적용의 예.

터베이스시스템에서는, Oracle SQL의 CREATE OPERATOR 문을 사용하여 두개의 필드를 하나로 머지하는 merge 연산자를 생성하고, merge 연산자의 입력 파라미터로서 처음 VARCHAR는 테이블 이름을, 다음 두개의 VARCHAR 는 머지될 필드 이름을 받도록 정의할 수 있다. 그리고 merge 연산의 실제 구현은 데이터 품질 관리 시스템이 구현한 f_merge 란 PL/SQL 프로스저나 함수로서 바인딩 할 수 있겠다 [9].

2.2. 매칭 연산자

매칭 연산자는 두개의 필드를 입력으로 받아 이들 필드의 값이 얼마나 근접한지를 계산하는 연산자이다. 전통적인 정보 검색(information retrieval) 분야에서는 거리 함수(distance function)란 개념을 사용해 정보간에 유사성을 계산해 냈는데, 데이터의 무균일과 중복을 처리하는 데이터 품질에서도 매칭 연산은 빠질 수 없는 연산자이다.

11. 매칭(maching) 연산자

정의: $\psi(R,i,j,S,k,l) = dis\tan ce(a_{ij}, b_{kl})$

테이블 R의 i번째 레코드의 j번째 필드 값인 a_{ij}와 테이블 S의 k번째 레코드의 l번째 필드 b_{kl}값의 유사성을 구하기 위해 distance 함수를 적용시킨다. distance 함수의 구현에는 적용 필드의 레코드들을 어떻게 한정할 것인지에 대한 사항도 명확해야 한다. 필드값 간의 매칭이 아닌 레코드간의 매칭이 목적인 분야에서는 매칭 연산자의 입력이 필드대신 레코드 단위로 정의될 수 있다 [11].

2.3. 통합 연산자

필드 값에 대한 매칭이 끝나면 매칭 결과에 따라 여러 데이터를 분류한 후, 과연 데이터들이 동일한 것인지를 판단하게 된다. 여러 데이터들이 동일한 경우로 판단되면 이들은 하나의 대표 값으로 통합하여 줄 수 있는데, 이때 필요한 연산자가 통합 연산자(integration operator for data quality)이다. 통합에는 먼저 데이터들을 분류하는 클래스화 혹은 클러스터링 과정과, 같은 분류에 속한 데이터들을 대표 값으로 바꾸어주는 과정이 필요하다. 참고로 기계학습(machine learning)에서는 데이터를 묶어 놓는 방식을 감독학습(supervised-learning)인지에 따라 감

독(supervised)인 경우를 클래스화(classification)이라 하고, unsupervised인 경우를 클러스터링(clustering)이라고 부른다 [6].

12 클래스화/ 클러스터링 (classification) 연산자

정의: $\chi(R,i)=j$

테이블 R의 i번째 필드 a_i는 클러스터 C_j에 속한다.

3. PRIMAL 매칭 알고리즘

매칭함수 $\psi(R,i,j,S,k,l)=dis\tan ce(a_{ij},b_{kl})$는 두 테이블 R, S로부터의 각 데이터 쌍에 대해 거리값(distance value)을 구해낸다. 매칭과정에서는 이 거리값을 사용하여 두개의 데이터가 동일한지(거리값이 0인 경우), 혹은 매우 비슷한지(거리값이 허용범위 ϵ보다 작은 경우)를 판정해 낸다.

데이터 셋이 매우 큰 경우라면 각 데이터에 매칭함수를 적용 시키는 것은 많은 비용(cost)이 든다. 따라서 대부분의 매칭 기법에는 어떻게 하면 매칭 비용을 적게 할 것인지를 고려하는데, 일반적으로 다음의 두 가지 방향에서 그 해법을 구한다.

1) 비용이 싼 매칭함수를 통한 필터링
2) 매칭 대상 영역의 축소

본 논문에서 제시하는 매칭 알고리즘 PRIMAL(a PRactical Matching ALgorithm) 역시 두 가지 방향에서 모두 접근하는데, 각 부분에 대해 PRIMAL이 어떻게 대처하고 있는지 살펴보기로 하자.

3.1 비용이 싼 매칭함수를 통한 필터링

테이블 R의 j번째 컬럼과 테이블 S의 l번째 컬럼 값들에 매칭함수 distance를 사용하여 얻어진 결과 집합 M은 다음과 같다.

$M=\{(a_{ij},b_{kl})\mid a_{ij}\in R,b_{kl}\in S,dis\tan ce(a_{ij},b_{kl})\le\varepsilon\}$, 여기서 a_{ij}는 R의 i번째 레코드의 j번째 컬럼이고 b_{kl}은 S의 k번째 레코드의 l번째 컬럼이다. 당연히 M은 $\Pi_j R\times\Pi_l S$의 부분 집합이 된다.

비용이 싼 또 다른 매칭함수를 통한 필터링 기법의 아이디어란 아래와 같은 성질을 가지는 매칭함수 distance' 와 매핑 함수 f 를 찾아내는 것이다.

1) For all $a_{ij},b_{kl},a_{ij}\in R,b_{kl}\in S$, $dis\tan ce'(f(a_{ij}),f(b_{kl}))\le dis\tan ce(a_{ij},b_{kl})$
2) 함수를 적용해 값을 구하는 비용을 $distance_{cost}$, f_{cost}, $distance'_{cost}$로서 나타낼 때,

$$\left(\sum_j f_{cost}(a_{ij})+\sum_l f_{cost}(b_{kl})\right)+\sum_{jl} dis\tan ce'_{cost}(f(a_{ij}),f(b_{kl}))\le\sum_{jl} dis\tan ce_{cost}(a_{ij},b_{kl})$$

1) 에서 $dis\tan ce'(f(a_{ij}),f(b_{kl}))\le dis\tan ce(a_{ij},b_{kl})\le\varepsilon$ 이므로, 이는 거짓-매치(false-matches)는 허용하지만, 거짓-디스미살(false-dismissal)은 허용하지 않음을 의미한다. 거짓-매치란 매핑함수 f를 미리 적용함으로 인해 실제로는 매치되는 a_{ij}, b_{kl}들이 $dis\tan ce'(f(a_{ij}),f(b_{kl}))>dis\tan ce(a_{ij},b_{kl})$ 이어서 결과로서 나타나지 않음을 말한다. 이와 대비되는 개념인 거짓-디스미살은 일단 매핑함수 f와 distance' 함수에 의해 걸러진 a_{ij}, b_{kl}들이 실

제 distance 함수를 적용하면 ϵ 보다 큰 값이 된다는 것을 의미한다. 당연히 이러한 거짓-디스미살은 일어나지 않게 된다.

2) 는 매핑함수 f를 적용하고 distance' 을 적용하는 근본 목적은, distance 함수를 적용하는 비용이 너무 크므로 이 보다 적은 비용을 가진 필터링 기법만이 의미있음을 말한다.

예를 들어 X, Y 두개의 학교명을 매칭하는 함수 distance를 Damerau-Levenshtein 메트릭 [2,5] 을 이용하여 다음과 같이 정의한다 하자.

문자열 X와 Y를 문자단위로 살펴 $X = x_1 x_2 x_3 \cdots x_n$, $Y = y_1 y_2 y_3 \cdots y_m$ 이라 하고, λ는 널(null) 문자를 뜻한다 하자. 그리고 $1 \le p \le n$, $1 \le q \le m$ 인 p, q에 대해 $\#S_i$, $\#D_i$, $\#I_i$를 Y를 X문자열과 동일하게 만들기 위하여 필요한 연산을 i번 수행하는데 필요한 치환 연산중에서, 몇 번의 문자열 치환(substitute) 연산이 나타나는지, 몇번의 제거(delete) 연산이 나타나는지, 몇번의 삽입(insert) 연산이 나타나는지를 나타낸다 하자. 다시 말해 #Si, #Di, #Ii 는 다음과 같이 정의 된다.

$\#S_i$: 총 i 번 연산중에서 $y_q \rightarrow x_p$ 로 만드는 치환(subsititute) 연산의 빈도 수

$\#D_i$: 총 i 번 연산중에서 $y_q \rightarrow \lambda$로 만드는 제거(delete) 연산의 빈도 수

$\#I_i$: 총 i 번 연산중에서 $\lambda \rightarrow x_p$ 로 만드는 삽입(insert) 연산의 빈도 수

그러면 문자열 X와 Y의 매칭함수는 치환, 제거, 삽입 연산의 중요도(weight)가 똑 같다고 가정했을 경우, 문자열 Y를 X로 만들기 위해 필요한 최소한의 연산으로 정의할 수 있다.

$$distance(X,Y) = \min_i(\#S_i + \#D_i + \#I_i)$$

예를 들어 두개의 학교명 문자열의 X와 Y의 값이 각각 "숙명여자대학교" 와 "숙명 어대" 이라고 하자. 주의할 것은 Y에는 "숙명" 과 "어대" 사이에 스페이스가 있고 실제 값일 "여대" 대신에 오타인 "어대"로 값이 되어 있다는 것이다. 이러한 문자열의 매칭함수 값은, 다시 말해 "숙명 어대"를 "숙명여자대학교" 라는 문자열과 같게 만들기 위해 필요한 최소 개수의 치환/제거/삽입 연산의 합이 몇 개인지를 말한다.

$$distance(\text{"숙명여자대학교"}, \text{"숙명 어대"}) = \\ = \min_i(\#S_i + \#D_i + \#I_i)$$

이 경우 i가 5인 경우, 치환 연산으로서 "어" →"여", 제거 연산으로서 " "→λ, 그리고 삽입 연산으로서 λ→"자", λ→"학", λ→"교" 가 필요한 것이 최소 숫자 i 가 되게 된다. 따라서 distance("숙명여자대학교", "숙명 어대")는 5가 되는 것이다.

이러한 distance 함수값은 Dynamic Programming 알고리즘 기법을 써서 구할 수 있는데[9], 이 경우 알고리즘의 복잡도는 X, Y 문자열의 길이가 위에서 나타낸 바와 같이 m, n인 경우 O(mn)이 되게 된다.

위의 distance 함수에 대해 매핑함수 f를 문자열의 크기를 나타내는 strlen()를 사용하고, 필터링함수 distance'으로서 두 문자열의 크기 차이로 다음과 같이 정의 할 수 있겠다.

$$distance'(X, Y) = abs\ (\ strlen(X), strlen(Y)\ )$$

원래 함수 distance가 문자열 Y를 X로 바꾸는데 필요한 최소한의 치환,삽입,제거 연산의 합이므로 위에서 정의된 f와 distance'은 (1)의 성질을 만족하게 된다.

3.2 매칭 대상 영역의 축소

이 기법은 매칭결과 집합 M의 원소가 될 것이라고 추정되는 레코드들만을 매칭함수를 적용시키는 방법이다. 다시 말해 매칭하는 대상 영역을 축소시키는 방법인 것이다. 테이블 R의 j번째 컬럼과 테이블 S의 l번째 컬럼 값들에 매칭할 경우 매칭될 수 있는 전체 집합은 $\Pi_j R \times \Pi_l S$ 이 된다. 그런데 실제로 매칭결과가 기대되는 후보자(candidates)들이 R에서 cand_x 프레디킷에 의해 정의되는 영역에 속하고 S 에서는 cand_y에 의해 정의되는 용역에 속한다면, $\sigma_{cand_x}(\Pi_j R) \times \sigma_{cand_y}(\Pi_l S)$ 에 대해서만 매칭함수를 적용 시켜도 되는 것이다.

예를 들어 두개의 문자열을 매칭하는데 있어 서로 실제로 같거나 매우 유사한, 다시 말해 매칭결과의 원소로 되는 데이터들은, 레코드들을 그 문자열 필드에 대해 정렬했을때 근처에 위치하게 되는 경향이 높다. 정열 후 레코드들을 일정 영역으로 나누어 인근 영역에 속하는 레코드들끼리만 매칭함수를 적용하는 방식이 소위 말하는 윈도우(window), 정렬-이웃(sorted-neighborhood), 혹은 이웃-조인

```
Algorithm PRIMAL
Input: a_ij, b_kl, a_ij∈R, b_kl∈S // 매치될 필드는 R의 i 번째 필드, S의 j번째 필드
Output: M 매칭 결과 집합
// distance, f, distance'는 다음과 같이 정의한다.
distance(X, Y) : min_i(#S_i+#D_i+#I_i)
f(X) : strlen(X) // 문자열 X의 길이
distance'(X, Y) : abs( f(X), f(Y) ), 즉 abs (strlen(X), strlen(Y))
M = {}
R 의 i 번째 필드값을 소팅한 후 'ㄱ', 'ㄴ', .. 등 동일한 한글 자순을 가진 레코드 집합으로 총 13개
로 파티션하여 각각을 Window1_i, 1≤i≤13 (한글자음수=13)이라 부르자. S 역시 같은 방법으로 파티션
하여 각 파티션을 Window2_i, 1≤i≤13 이라 부르자.
for i=1 .. 13
  for j=1 .. 13
    Cand = {(p1,p2)| distance'(f(p1), f(p2))<ε, p1∈Window1_i, p2∈Window2_j};
      for each element (p1, p2) ∈ Cand
        if distance(p1, p2) < ε then
          M = M ∪ {(p1,p2)}; end if
  end for j
end for i
return M;
```

〈그림 6〉 PRIMAL 매칭 알고리즘

(neighborhood-join)이라 불리는 방식이다.

이들 방식은 매칭 대상 영역, 즉 윈도우를 어떻게 잡는가가 문제의 핵심이 된다. 참고로 이를 거짓-매치(false-matches)와 거짓-디스미살(false-dismissal) 관점에서 살펴보면, 거짓-매치 와 거짓-디스미살이 모두 발생할 수 있다.

이 밖에도, 하나 이상의 필드들을 매칭 하고 이들 간의 closure 연산을 통해 두개의 레코들의 매칭을 구해나가는 다중패스-이웃-조인(multipass-neighborhood-join) 방식이 있는데, 우리의 경우에서는 레코드들간의 매칭이 아닌 두 필드 만의 매칭을 고려하므로 이용하지 않았음을 밝혀 둔다.

이상에서 살펴본 1) 매칭함수를 통한 필터링 기법과 2) 매칭 대상 영역의 축소 기법은 방법론적인 구별에 불과하며, 실제로 사용되는 알고리즘에서는 이 두 가지 모두를 사용한다. 예를 들어 다음 그림 6은 학교명 문자열 필드를 매칭하기 위하여 본 논문에 제시하는 PRIMAL 매칭 알고리즘을 슈도코드로 소개한다. 그림에서는 우리가 사용하는 distance 함수, distance' 필터링 함수, f 매핑 함수를 이웃-조인(neighborhood-join) 방식의 매칭 대상 영역 축소에 적용한 보여주고 있다.

4. 관련 연구

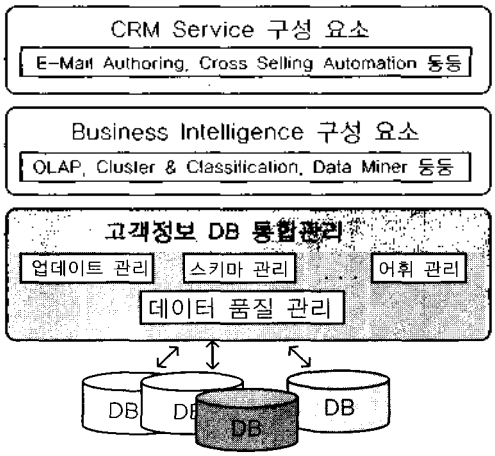
기업이 이러한 중복되고 무균일한 정보에 바탕을 두고 기업활동을 한다면 단순한 기업 활동에서부터 중요한 의사 결정에까지 그 효율성 및 경제성에 지대한 악영향을 미칠 수 있게 된다 [7,10,12]. 따라서 정보의 일치성과 중복관리를 체계적이고 효율적으로 관리 해 줄 수 있는 데이터 품질 관리 시스템이 요구되는데, 현재 정보 기술의 선두국인 미국과 일부 유럽 국가들에서는 이에 대한 연구가 대학 및 연구소, 그리고 기업체간에 활발히 이루어지고 있다 [3,8,11]. 특히 데이터웨어하우싱과 데이터마이닝분야에서 최근 몇 년 안에 이루어진 연구 및 실제 개발 경험들은 데이터 품질 관리 시스템에 직접적으로 영향을 주며 상호 발전관계를 형성하고 있다 [9,11].

레코드 매칭기법은 바로 클러스터빌더와 룰생성기 및 최적화기(Rule Generator and Optimizer)에서 필수적으로 사용되는 알고리즘으로, 그 성능은 전체 품질관리 시스템의 성능에 지대한 영향을 준다. 대용량 데이터베이스에 레코드 값의 효율적인 중복 처리 문제는 전통적인 물리적 데이터베이스 분야에서도 일찍이 다루어졌고 고전적인 연구로서 간단한 중첩 루프를 통한 블록 연산을 들 수 있다 [1]. 레코드 매칭에 관한 기존의 연구는 최근 들어 데이터마이닝 분야의 데이터 클러스터링에 주로 나타난다. 데이터 클러스터링이란 어느 데이터끼리 서로 관련을 지어서 그룹핑을 하는 문제인데, 이를 크게 나누어보면 계층적(hierarchical) 클러스터링과 분할적(partitioning) 클러스터링이 있다. 이 밖에도 Baysian과 같은 수학 통계적인 방식을 쓰거나, 중력(gravity)을 이용한 방식, 혹은 인스턴스에 기반해 데이터를 분류하는 기법 등이 있다 [6,7].

데이터 품질관리와 CRM의 연관 연구는

[7,10,4] 등에서 소개되고 있다. 특히 [4]에서는 일반적인 CRM 시스템 아키텍쳐를 CRM Service, Business Intelligence, 고객정보 DB 통합관리 의 세 계층으로 제시하였는데, 데이터 품질관리는 고객정보 DB 통합관리 의 업데이트 관리, 스키마 관리, 어휘 관리등과 상호 연동 혹은 포함 관계를 이루는 아키텍쳐 구성부분이라 할 수 있겠다. (그림 7)

최근들어 데이터 클리닝에 대한 관심이 높아지면서 데이터 클리닝 프로토타입 시스템이 등장하고 있다 [3, 8, 9]. [3]는 데이터 클리닝 시스템을 논리적 비유와 물리적 비유의 두가지 단계로 나누어 설계하고, 논리적 비유에 필요한 연산자를 정의한다. [8,9]는 사용자 인터페이스를 중심으로 데이터 클리닝용 시스템 브라우저에 다양한 연산자가 실제로 어떻게 구현될 수 있는지 소개하고 있다. 이들 연구는 본 논문의 기초연구가 된다 할 수 있는데, 본 논문에서는 더 나아가 한글 데이터 매칭 등을 위한 PRIMAL 알고리즘을 제시하



〈그림 7〉 일반적인 CRM 아키텍쳐와 데이터 품질관리

였다는 점이 다르다. 또 실제 시스템에 필요한 연산자의 그룹핑 역시 [3]에 비해 확장되어 있다.

5. 결론 및 연구 방향

데이터 품질 관리 시스템(data quality management system)은 복잡한 다양한 구성요소로 이루어지는 시스템이다. 본 논문에서는 현실적으로 가능하고 추후 그 연구업적이 계속적으로 연계될 수 있도록 먼저 그 기초연구인, 필요한 연산자의 정의 및 분류, 프로토타입핑 구현에 필요한 적당한 매칭 알고리즘의 제시등에 국한하였다. 그리고 데이터 클러스터 연구에서는 기존의 클러스터링 기법 특히 데이터 마이닝 분야에서의 클러스터링 기법들이 구체적으로 데이터 품질 관리에서는 어떻게 응용될 수 있으며, 실제적으로 사용될 수 있는 측면으로서 튜닝(tuning) 기법들이 연구 중점 됨으로 해서 데이터베이스 연구계 및 기업체에 그 활용성이 기대된다.

데이터 클리닝과정에서 야기될 수 있는 스키마 변환은 데이터베이스 응용 프로그램에 영향을 미칠 수 있는데, 영향에 대한 분석은 좋은 연구 방향이 된다. 데이터의 독립성이 유지되어지게 데이터베이스와 응용프로그램이 설계 구현된 경우라면 스키마 변환에 따른 영향이 미미하겠지만 CRM이나 기타 웹기반 전자상거래시스템과 같이 많은 ASP 프로그램들이 데이터베이스 스키마에 종속되어지게 구현된 경우라면 문제는 심각하게 된다. 이러한 경우 데이터 품질 관리는 데이터베이스 내

부적인 품질 관리 뿐 아니라 이로 인한 파급 효과도 고려할 필요성이 있게 된다.

참 고 문 헌

[1] Dina Bitton, David J. DeWitt: Duplicate Record Elimination in Large Data Files. *ACM Transactions on Database Systems, 8(2)*, 1983, ACM Press.

[2] Fred Damerau, A Technique for Computer Detection and Correcting of Spelling Errors, *Communications of the ACM 7(3)*, 1964, ACM Press.

[3] Helena Galhardas, Daniela Florescu, Dennis Shasha, Eric Simon, Cristian-Augustin Saita: Declarative Data Cleaning: Language, Model, and Algorithms. *Proc. of the International Conference on Very Large Data Bases*, 2001, Morgan Kaufmann.

[4] Han-joon. Kim, Taehee Lee, Sang-goo Lee, and Jonghun Chun, Automated Data Warehousing for Rule-based CRM Systems, *Proc. of the 14th Australasian Database Conference*, Australian Computer Society, Feb 2003.

[5] Vladimir Levenshtein, Binary Codes Capable of Correcting Detections, Insertions, and Reversals, *Soviet Physics Doklady 10(8)*, 1966.

[6] J. Ross Quinlan, Induction of Decision Trees, *Machine Learning 1(1)*, 1986, Kluwer Academic Publisher.

[7] Erhard Rahm, Hong Hai Do: Data Cleaning: Problems and Current Approaches. *IEEE Data Engineering Bulletin 23(4)*, 2000, IEEE Computer Society.

[8] Vijayshankar Raman, Joseph M. Hellerstein: Potter's Wheel: An Interactive Data Cleaning System. *Proc. of the International Conference on Very Large Data Bases*, 2001, Morgan Kaufmann.

[9] Junho Shim, Vassilios S. Verykios, D2Q Project, *Technical Report (progress report)*, December 2000, Drexel University.

[10] Giri K. Tayi et al, Examining Data Quality, *Communications of the ACM 41(2)*, 1998, ACM Press.

[11] Vassilios S. Verykios, Mohamed G. Elfeky, Ahmed K. Elmagarmid, Munir Cochinwala, and Sid Dalal, On the Accuracy and Completeness of the Record Matching Process, *International Conference on Information Quality*, 2000, MIT Press.

[12] Richard Y. Wang, Veda C. Storey, Christopher P. Firth: A Framework for Analysis of Data Quality Research. *IEEE Transactions on Knowledge and Data Engineering 7(4)*, 1995, IEEE Computer Society.

저 자 소 개

심준호 (E-mail : jshim@sookmyung.ac.kr)

1990. 2. 서울대학교 계산통계학과(학사)

1994. 2. 서울대학교 계산통계학과 전산과학전공(이학석사)

1998. 12. Northwestern Univ., Illinois, USA., Electrical & Computer Engineering, 공학박사

1999. 2 ~ 1999. 11. Computer Associates Int'l, New York, USA, R&D Staff

1999. 11 ~ 2001. 2. Drexel Univ., Pennsylvania, USA, Assistant Prof.

2001. 3 ~ 현재 숙명여자대학교 정보과학부 조교수

관심 분야 : 데이타베이스, 데이타웨어하우스, 전자상거래, 웹