TITLE:

CS-78-34

AUTHOR:

NAME:	DATE TAKEN	DATE RETURNED	SIGNATURE
Frank Tompa	June 20/80	June 25/80	
"	July 14/80	July 14/80	

AN IMPROVED THIRD NORMAL FORM FOR RELATIONAL DATA BASES

Tok-Wang Ling †
Frank Wm. Tompa
Tiko Kameda ††

Department of Computer Science
University of Waterloo
Waterloo, Ontario, Canada
N2L 3G1

ABSTRACT

In this paper, we show that some Codd third normal form relations may contain "superfluous" attributes because the definitions of transitive dependency and prime attribute are inadequate when applied to sets of relations. To correct this, an improved third normal form is defined and an algorithm is given to construct a set of relations from a given set of functional dependencies in such a way that the superfluous attributes are guaranteed to be removed. This new normal form is compared with other existing definitions of third normal form, and the deletion normalization method proposed is shown to subsume the decomposition method of normalization.

Key Words and Phrases: data base design, relational schema, functional dependency, transitive dependency, prime attribute, third normal form, normalization, covering, reconstructibility.

CR *Categories:* 3.70, 4.33

This paper has been accepted for publication in *Transactions on Database Systems*.

July 12, 1980

† Current address: Department of Computer Science, Nanyang University, Upper Jurong Road, Singapore 22
†† Department of Electrical Engineering

AN IMPROVED THIRD NORMAL FORM FOR RELATIONAL DATA BASES

Tok-Wang Ling †
Frank Wm. Tompa
Tiko Kameda ††

Department of Computer Science
University of Waterloo
Waterloo, Ontario, Canada
N2L 3G1

1. Introduction.

Several years ago the relational model was introduced in order that data base design could be grounded in a well-established mathematical discipline. The basic notion of a relation was augmented by the concepts of functional dependencies and normal forms in an attempt to provide integrity by reducing undesirable updating anomalies [Codd 71]. A particularly undesirable form of redundancy is the presence in a relation of an attribute whose value can always be derived from other attributes (perhaps using other relations) and whose value is not needed to derive other attributes' values. Such an attribute will be called *superfluous* in that relation; a formal definition will be given in Section 4.

In this paper, we give examples to show that some Codd third normal form relations and Boyce-Codd normal form relations [Codd 74] may contain some superfluous attributes because the definitions of transitive dependency and prime attribute are inadequate when applied to a set of relations.

To correct this, we give new definitions to replace the concepts of transitive dependency and prime attribute. A normal form for a set of relations is then defined based on these new definitions. Since the old definitions are inadequate, all existing normalization methods based on them must be re-evaluated in the light of the new definitions. We present the deletion normalization algorithm which is more powerful than the decomposition method, and we show that the set of relations produced by this algorithm contains no superfluous attributes.

2. The relational model

A relational data base, consisting of several interrelated relations, was first introduced by Codd [Codd 70]. A relation is defined as follows: given sets of atomic (non-decomposable) elements $DOM_1, DOM_2, \ldots, DOM_n$ (not necessarily distinct), **T** is a *first normal form relation* (or simply *relation*) on these n sets if it is a set of ordered n-tuples $(D_1, D_2, \ldots, D_n)$ such that D_i belongs to DOM_i for $i=1,2,\ldots,n$. Thus $\mathbf{T} \subseteq DOM_1 \times DOM_2 \times \cdots \times DOM_n$, where $\times$ denotes the Cartesian product. $DOM_1, DOM_2, \ldots, DOM_n$ are called the *domains* of **T**. Rather than referencing each use of a domain by position number, each is assigned a unique role name, called an *attribute* of **T**. For any tuple in **T**, the value for the attribute named B is referred to as a B-value, and, for a set of attributes $X = \{B_1, B_2, \ldots, B_p\}$, the tuple's values for the attributes in X is referred to as an X-value; the values of the other attributes in that tuple are said to be *associated*

† Current address: Department of Computer Science, Nanyang University, Upper Jurong Road, Singapore 22
†† Department of Electrical Engineering

The work reported in this paper was partially supported by the Natural Sciences and Engineering Research Council of Canada under grants A9292 and A4315.

with that X-value. A set of attributes Y of **T** is said to be *functionally dependent* on a set of attributes X of **T** if each X-value in **T** has associated with it exactly one Y-value in **T** (at any time). This is denoted by $X \rightarrow Y$ and is called a *functional dependency* of **T**; X and Y are termed the *left* and *right sides* of the dependency, respectively.

The relational algebra originally proposed by Codd includes several operations for manipulating relations. Of particular interest here are the operations of projection and (natural) join, defined [illegible]

[illegible]

[illegible] for some T [illegible] $\mathbf{T}_1$ and T [illegible] $\mathbf{T}_2$, [illegible]

where · denotes catenation (and possibly reordering of attributes).

It is important to realize that as data occurrences are inserted, deleted, and modified in a data base, the relations (that is, the sets of tuples) in that data base are altered. However, the relational algebra does not include facilities for altering a relation's set of attributes nor its set of functional dependencies. Thus these two sets are time-invariant properties associated with the *relation scheme* **R** which serves as a framework for a time-varying sequence of relations **T**. Henceforth the notions of projection and join, when applied to relation schemes, refer [illegible]

[illegible]

[illegible] functional d[illegible] **R** [illegible] addition to [illegible] functional dependencies [illegible] 74 [illegible]

[illegible] if $Y \subseteq X$ [illegible] $X \rightarrow Y \in F^+$

[illegible] and Z of [illegible] $\rightarrow Y \in F^+$ [illegible]

[illegible] and Z of [illegible] $X \rightarrow Y \in F^+$ [illegible]

[illegible] and a [illegible] dependencies F, a functional dependency $X \rightarrow B \in F^+$, where $X \subseteq A$ and $B \in A$, is said to be a *full dependency* of **R** (or B is *fully dependent* on X under F) if there exists no proper subset $X' \subset X$ such that $X' \rightarrow B \in F^+$. Two sets of attributes $X \subseteq A$ and $Y \subseteq A$ are said to be *functionally equivalent* (or simply *equivalent*) if $X \rightarrow Y \in F^+$ and $Y \rightarrow X \in F^+$. X and Y are said to be *properly functionally equivalent* (or simply *properly equivalent*) if X and Y are equivalent and there exist no proper subsets $X' \subset X$ and $Y' \subset Y$ such that $X' \rightarrow Y \in F^+$ or $Y' \rightarrow X \in F^+$. An attribute B is said to be *transitively dependent* on $X \subseteq A$ if there exits $Y \subset A$ such that $B \in A - Y$, $X \rightarrow Y \in F^+$, $Y \rightarrow B \in F^+$, and $Y \rightarrow X \notin F^+$.

For a relation scheme **R** having a set of attributes A and a set of functional dependencies F, a set of attributes $K \subseteq A$ is called a *candidate key* (or simply *key*) of **R** (or, colloquially, a key for A) if $K \rightarrow A \in F^+$ and for all $X \subset K$, $X \rightarrow A \notin F^+$. It is easy to prove that every relation has at least one key and that some may have more than one key. An attribute in A is called a *prime attribute* of **R** if it is contained in some key of **R**. All other attributes in A are called non-prime attributes.

Codd recognized immediately that certain relation schemes may contain some redundancy. Consider, for example, the relation in Figure 1a which represents stock information for some hypothetical manufacturer:

MODEL#	SERIAL#	PRICE	COLOUR	NAME	YEAR
1234	342	13.25	blue	pot	1974
1234	347	13.25	red	pot	1974
1234	410	14.23	red	pot	1975
1465	347	9.45	black	pan	1974
1465	390	9.82	black	pan	1976
1465	392	9.82	red	pan	1976
1465	401	9.82	red	pan	1976
1465	409	9.82	blue	pan	1976
1623	311	22.34	blue	kettle	1973
1623	390	30.21	blue	kettle	1976
1623	410	28.55	black	kettle	1975
1623	423	28.55	black	kettle	1975
1623	428	28.55	blue	kettle	1975
1654	435	28.55	red	kettle	1975

Figure 1a. Stock inventory universal relation

NAME	YEAR	PRICE
pot	1974	13.25
pot	1975	14.23
pan	1974	9.45
pan	1976	9.82
kettle	1973	22.34
kettle	1975	28.55
kettle	1976	30.21

Figure 1b. Stock inventory price relation

For each stock item, the model number, serial number, list price, colour, model name, and year of manufacture for an article are entered. The price and colour are unique for a given model number and serial number. If it is further assumed that the model name can be determined from the model number, the year of manufacture can be determined from the serial number, and the price can be determined from the model name and the year, then the set of functional dependencies is

$$\{ \quad \{MODEL\#, SERIAL\#\} \rightarrow \{PRICE, COLOUR\}, \quad \{MODEL\#\} \rightarrow \{NAME\}, \quad \{SERIAL\#\} \rightarrow \{YEAR\}, \quad \{NAME, YEAR\} \rightarrow \{PRICE\} \quad \}$$

If a new model for some year is announced but no items for that model and year are yet in stock (and therefore no model number and serial number are yet available), then the price information cannot be entered for that model (i.e., NAME) and year (the use of null or undefined values in other fields could cause problems [Osborn 77]). This is called the *insertion anomaly*. Now if the last item of stock for a particular model and year is sold and therefore a tuple with a {NAME,YEAR}-value that appeared in this tuple only is deleted, then the price information for this {NAME,YEAR}-value would be lost. This is called the *deletion anomaly*. If the PRICE-value for a {NAME,YEAR}-value were to be changed then that attribute's values for all tuples that have this given {NAME,YEAR}-value would also have to be changed to maintain the consistency of the data base. This is called the *rewriting anomaly*. Now suppose that the data base contains another relation containing all the model name, year, and price information as in Figure 1b. In this case, the superfluous attribute PRICE could be removed from the universal stock relation scheme without losing any information from the data base, whereas it would not be removable from the stock inventory price scheme without re-introducing anomolies.

One process that attempts to remove undesirable updating anomalies and redundant attributes from the relation schemes is called normalization, which was originally defined in two stages [Codd 71]. A (first normal form) relation scheme **R** is in *second normal form* if every non-prime attribute of **R** is fully dependent on each key of **R**. A relation scheme **R** is in *Codd third normal form* if it is in second normal form and each non-prime attribute of **R** is not transitively dependent on every key of **R**. For example, the set of relations in third normal form in Figure 2 maintains the same data as the stock inventory depicted in Figure 1.

MODEL#	SERIAL#	PRICE	COLOUR
1234	342	13.25	blue
1234	347	13.25	red
1234	410	14.23	red
1465	347	9.45	black
1465	390	9.82	black
1465	392	9.82	red
1465	401	9.82	red
1465	409	9.82	blue
1623	311	22.34	blue
1623	390	30.21	blue
1623	410	28.55	black
1623	423	28.55	black
1623	428	28.55	blue
1654	435	28.55	red

SERIAL#	YEAR
311	1973
342	1974
347	1974
390	1976
392	1976
401	1976
409	1976
410	1975
423	1975
428	1975
435	1975

MODEL#	NAME
1234	pot
1465	pan
1623	kettle
1654	kettle

NAME	YEAR	PRICE
pot	1974	13.25
pot	1975	14.23
pan	1974	9.45
pan	1976	9.82
kettle	1973	22.34
kettle	1975	28.55
kettle	1976	30.21

Figure 2. Stock inventory normalized relations

THEOREM 1. A relation scheme **R** is in Codd third normal form if and only if each non-prime attribute is not transitively dependent on an arbitrarily chosen key of **R**.

PROOF: The proof is based on the following two lemmas which show that a non-full or transitive dependency of an attribute on any one key of **R** implies the transitive dependency of that attribute on all keys of **R**.

LEMMA 1.1 Let **R** be a relation scheme consisting of a set of attributes A and a set of functional dependencies F and let $\mathrm{B} \in A$. If there exists a key K of **R** with $\mathrm{B} \notin \mathrm{K}$ and $\mathrm{K} \rightarrow \mathrm{B}$ is not a full dependency, then B is transitively dependent on all keys of **R**.

PROOF: Since $\mathrm{B} \notin \mathrm{K}$ and $\mathrm{K} \rightarrow \mathrm{B}$ is not a full dependency, therefore there exists a proper subset $\mathrm{X} \subset \mathrm{K}$ such that $\mathrm{B} \notin \mathrm{X}$ and $\mathrm{X} \rightarrow \mathrm{B} \in F^+$. Since K is a key and X is a proper subset of K, $\mathrm{X} \rightarrow \mathrm{K} \notin F^+$. Now for any key K' of **R**, $\mathrm{K}' \rightarrow \mathrm{X} \in F^+$, $\mathrm{X} \rightarrow \mathrm{K}' \notin F^+$, $\mathrm{X} \rightarrow \mathrm{B} \in F^+$, and $\mathrm{B} \notin \mathrm{X}$. Hence B is transitively dependent on K', which proves the lemma.

LEMMA 1.2 Let $\mathbf{R}$ be a relation scheme consisting of a set of attributes A and a set of functional dependencies F and let $B \in A$. If there exists a key K of $\mathbf{R}$ such that B is transitively dependent on K, then B is also transitively dependent on all keys of $\mathbf{R}$.

PROOF: Let K$'$ be any other key of $\mathbf{R}$. By definition, $K' \rightarrow B \in F^+$. Now if B is transitively dependent on K, then there exists a set of attributes $X \subset A$ such that $K \rightarrow X \in F^+$, $X \rightarrow K \notin F^+$, $X \rightarrow B \in F^+$, and $B \notin X$. Hence $K' \rightarrow X \in F^+$, $X \rightarrow K' \notin F^+$, $X \rightarrow B \in F^+$, and $B \notin X$. Thus B is transitively dependent on K$'$, which proves the lemma.

Closely related to closure is the notion of derivability [Lucchessi 78], as follows: a set of attributes Y is *derivable* from a set of attributes X using the set of functional dependencies F if there exists a sequence of attribute sets (called a *derivation* of Y from X) $<X_0, X_1, X_2, \ldots, X_n>$ for $n \geqslant 0$ such that $X = X_0$, $Y \subseteq X_n$, and (unless $n = 0$) for i in the range 1 to n there exists a functional dependency $V \rightarrow W \in F$ such that $V \subseteq X_{i-1}$, $W \not\subseteq X_{i-1}$, and $X_i = X_{i-1} \cup W$.

THEOREM 2. [Lucchesi 78, Beeri 79] Given a set of attributes A, $X \subseteq A$ and $Y \subseteq A$, and a set of functional dependencies F defined on subsets of A, Y is derivable from X using F if and only if $X \rightarrow Y \in F^+$. Furthermore, since $B_{i-1} \subset B_i$ and $B_{i-1} \neq B_i$, derivability can be decided in $O(|F|\,|A|)$ time.†

Instead of considering the derivation of a particular set of attributes Y, it is sometimes convenient to find the set of all attributes derivable from X using F. A *maximal derivation* from a set of attributes X using a set of functional dependencies F is a sequence of attribute sets $<X_0, X_1, \ldots, X_n>$ for $n \geqslant 0$ such that $X = X_0$, for i in the range 1 to n there exists a functional dependency $V \rightarrow W \in F$ such that $V \subseteq X_{i-1}$ and $W \not\subseteq X_{i-1}$ and $X_i = X_{i-1} \cup W$, and there is no functional dependency $V' \rightarrow W' \in F$ such that $V' \subseteq X_n$ and $W' \not\subseteq X_n$. Lucchesi and Osborn have shown that the terminal set in a maximal derivation from X using F is independent of the particular derivation; henceforth X_n will be called the *closure of* X *relative to F.*

3. The problem of superfluous attributes

Given A, a set of attributes, and F, a set of functional dependencies among subsets of A, it is desirable to describe a set of relation schemes, henceforth called a *relational scheme*, $\mathbf{R} = \{\mathbf{R}_1, \ldots, \mathbf{R}_n\}$ such that the following three properties hold:

1. The relationships among the data values to be stored using $\mathbf{R}$ are equivalent to those that would be stored using a single relation scheme $\mathbf{R}_0$ involving all of A. That is, at all times, $\mathbf{R}_i = \mathbf{R}_0[A_i]$ where A_i is the set of attributes in $\mathbf{R}_i$, and $\mathbf{R}_1 * \mathbf{R}_2 * \cdots * \mathbf{R}_n = \mathbf{R}_0$.
2. The verification that a set of relations described by $\mathbf{R}$ (i.e., relational instances for $\mathbf{R}_1, \ldots, \mathbf{R}_n$) conforms to all functional dependencies in F requires only the examination of relations corresponding to $\mathbf{R}_1, \ldots, \mathbf{R}_n$ individually. Furthermore, the only functional dependencies that need be examined are ones for which the left side contains a key of some $\mathbf{R}_i$. It is required that $(\bigcup \{X \rightarrow A_i - X \mid X \subseteq A_i \text{ and } X \rightarrow A_i \in F^+\})^+ = F^+$.
3. Each relation scheme is free of redundant attributes, that is, those whose presence is not required for maintaining the other two properties. It should be noted here that the redundancy considered is with respect to given sets of attributes and functional dependencies only; the redundancy under consideration is thus not that which may arise from other time-invariant properties of the data base (see, for example, [Delobel 78]).

These goals have been defended elsewhere (see, for example, [Rissanen 77, Beeri 78, Biskup 79]) and have been termed *reconstructibility* (or losslessness), *covering*, and *normalization*, respectively.

Several characterizations for the elements of $\mathbf{R}$ have been given since the introduction of the relational model, but none have satisfactorily met the requirement of normalization in that redundant attributes are sometimes permitted. For the remainder of this section, we will demonstrate [illegible]

†For a set S, | S| denotes the cardinality of S, as opposed to the length of the description of S as in [Beeri 79].

LEMMA 1.2 Let **R** be a relation scheme consisting of a set of attributes A and a set of functional dependencies F and let $B \in A$. If there exists a key K of **R** such that B is transitively dependent on K, then B is also transitively dependent on all keys of **R**.

PROOF: Let K′ be any other key of **R**. By definition, $K' \rightarrow B \in F^+$. Now if B is transitively dependent on K, then there exists a set of attributes $X \subset A$ such that $K \rightarrow X \in F^+$, $X \rightarrow K \notin F^+$, $X \rightarrow B \in F^+$, and $B \notin X$. Hence $K' \rightarrow X \in F^+$, $X \rightarrow K' \notin F^+$, $X \rightarrow B \in F^+$, and $B \notin X$. Thus B is transitively dependent on K′, which proves the lemma.

Closely related to closure is the notion of derivability. [Lucchessi 78], as follows: a set of attributes Y is *derivable* from a set of attributes X using the set of functional dependencies F if there exists a sequence of attribute sets (called a *derivation* of Y from X) $<X_0, X_1, X_2, \ldots, X_n>$ for $n \geqslant 0$ such that $X = X_0$, $Y \subseteq X_n$, and (unless $n = 0$) for i in the range 1 to n there exists a functional dependency $V \rightarrow W \in F$ such that $V \subseteq X_{i-1}$, $W \not\subseteq X_{i-1}$, and $X_i = X_{i-1} \cup W$.

THEOREM 2. [Lucchesi 78, Beeri 79] Given a set of attributes A, $X \subseteq A$ and $Y \subseteq A$, and a set of functional dependencies F defined on subsets of A, Y is derivable from X using F if and only if $X \rightarrow Y \in F^+$. Furthermore, since $B_{i-1} \subset B_i$ and $B_{i-1} \neq B_i$, derivability can be decided in $O(|F|\,|A|)$ time.†

Instead of considering the derivation of a particular set of attributes Y, it is sometimes convenient to find the set of all attributes derivable from X using F. A *maximal derivation* from a set of attributes X using a set of functional dependencies F is a sequence of attribute sets $<X_0, X_1, \ldots, X_n>$ for $n \geqslant 0$ such that $X = X_0$, for i in the range 1 to n there exists a functional dependency $V \rightarrow W \in F$ such that $V \subseteq X_{i-1}$ and $W \not\subseteq X_{i-1}$ and $X_i = X_{i-1} \cup W$, and there is no functional dependency $V' \rightarrow W' \in F$ such that $V' \subseteq X_n$ and $W' \not\subseteq X_n$. Lucchesi and Osborn have shown that the terminal set in a maximal derivation from X using F is independent of the particular derivation; henceforth X_n will be called the *closure of* X *relative to F.*

3. The problem of superfluous attributes

Given A, a set of attributes, and F, a set of functional dependencies among subsets of A, it is desirable to describe a set of relation schemes, henceforth called a *relational scheme*, $\mathbf{R} = \{\mathbf{R}_1, \ldots, \mathbf{R}_n\}$ such that the following three properties hold:

1. The relationships among the data values to be stored using **R** are equivalent to those that would be stored using a single relation scheme $\mathbf{R}_0$ involving all of A. That is, at all times, $\mathbf{R}_i = \mathbf{R}_0[A_i]$ where A_i is the set of attributes in $\mathbf{R}_i$, and $\mathbf{R}_1 * \mathbf{R}_2 * \cdots * \mathbf{R}_n = \mathbf{R}_0$.
2. The verification that a set of relations described by **R** (i.e., relational instances for $\mathbf{R}_1, \ldots, \mathbf{R}_n$) conforms to all functional dependencies in F requires only the examination of relations corresponding to $\mathbf{R}_1, \ldots, \mathbf{R}_n$ individually. Furthermore, the only functional dependencies that need be examined are ones for which the left side contains a key of some $\mathbf{R}_i$. It is required that $(\bigcup \{X \rightarrow A_i - X \mid X \subseteq A_i \text{ and } X \rightarrow A_i \in F^+\})^+ = F^+$.
3. Each relation scheme is free of redundant attributes, that is, those whose presence is not required for maintaining the other two properties. It should be noted here that the redundancy considered is with respect to given sets of attributes and functional dependencies only: the redundancy under consideration is thus not that which may arise from other time-invariant properties of the data base (see, for example, [Delobel 78]).

These goals have been defended elsewhere (see, for example, [Rissanen 77, Beeri 78, Biskup 79]) and have been termed *reconstructibility* (or losslessness), *covering*, and *normalization*, respectively.

Several characterizations for the elements of **R** have been given since the introduction of the relational model, but none have satisfactorily met the requirement of normalization in that redundant attributes are sometimes permitted. For the remainder of this section, we will [illegible]

---

†For a set S, |S| denotes the cardinality of S, as opposed to the length of the description of S as in [Beeri 79].

[illegible] that Codd third normal form and Boyce-Codd normal form are inadequate normalization criteria.

EXAMPLE 1. Let A = ABCDEF † and F = {AB→CD, A→E, B→F, EF→C} and consider the relational schema $\mathbf{R}$={$\mathbf{R}_1$, $\mathbf{R}_2$, $\mathbf{R}_3$, $\mathbf{R}_4$} where $\mathbf{R}_1$ has attributes ABCD and key AB, $\mathbf{R}_2$ has attributes AE and key A, $\mathbf{R}_3$ has attributes BF and key B, and $\mathbf{R}_4$ has attributes EFC and key EF. This [illegible] models the relations depicted in Figure 2, where A is the model number, B [illegible], C the price, D the colour, E the model name, and F is the year of manufacture. [illegible] $\mathbf{R}$ has the properties of reconstructibility and covering and that each relation in $\mathbf{R}$ is in [illegible] normal form. In particular, it must be noted that Codd third normal form [illegible] scheme at a time. For example, considering $\mathbf{R}_1$, there exists no subset X [illegible] X→C∈F^+ and X→AB∉F^+, hence C is not transitively dependent on AB in $\mathbf{R}_1$. [illegible], AB→D is also not a transitive dependency in $\mathbf{R}_1$. Thus $\mathbf{R}_1$ is in Codd third normal form, as are $\mathbf{R}_2$, $\mathbf{R}_3$, and $\mathbf{R}_4$. Now consider an instance of $\mathbf{R}$ containing tuples (A_1, B_1, C_1, D_1) for $\mathbf{R}_1$, (A_1, E_1) for $\mathbf{R}_2$, (B_1, F_1) for $\mathbf{R}_3$, and (E_1, F_1, C_2) for $\mathbf{R}_4$. Here the AB-value A_1B_1 can derive the C-value C_2 by using relations other than the one corresponding to $\mathbf{R}_1$. If $C_1 \neq C_2$, then even if $\mathbf{R}_1$ satisfies the functional dependency AB→C, the data base will be inconsistent. Similarly, if the data base is consistent, the C-value for some tuple for $\mathbf{R}_1$ cannot be altered without checking in other relations that the data base will not become inconsistent; that is, this normalization has not eliminated the potential for updating anomalies. It is easy to show that although AB→C is not a transitive dependency in $\mathbf{R}_1$, C is a superfluous attribute, i.e., it can be deleted from the relation scheme $\mathbf{R}_1$ while still preserving the functional dependency AB→C in $\mathbf{R}$.

EXAMPLE 2. Let A = ABCDEF and F = {AD→B, B→C, C→D, AB→E, AC→F} and consider the relational schema $\mathbf{R}$={$\mathbf{R}_1$, $\mathbf{R}_2$, $\mathbf{R}_3$} where $\mathbf{R}_1$ has attributes ABCDEF and keys AB, AC, and AD; $\mathbf{R}_2$ has attributes BC and key B; and $\mathbf{R}_3$ has attributes CD and key C. Again $\mathbf{R}$ has the properties of reconstructibility and covering, and each relation in $\mathbf{R}$ is in Codd third normal form. In this case, the attribute C is superfluous in $\mathbf{R}_1$, and, although it is a prime attribute and [illegible] not even considered for transitive dependencies when constructing Codd third normal form, C can (and should) be dropped from $\mathbf{R}_1$ while still preserving all functional dependencies in which it is involved.

From these two examples it is clear that the definitions of transitive dependency and prime attribute as applied to defining Codd third normal form are inadequate for describing relations [illegible] of "superfluous" attributes. Since the definitions are inadequate, any normalization [illegible] based on them may also be inadequate. For example, Bernstein's method [Bernstein 76] [illegible] fact, produce sets of relation schemes as in Example 2; that is, the method [illegible] produce schemes that are free of "superfluous" attributes.

[illegible] that Codd third normal form did still permit some anomalies, a [illegible] by Kent and by Boyce and Codd [Kent 73, Codd 74] [illegible] scheme $\mathbf{R}$ is in *Boyce-Codd normal form* if (it is in first normal form and) for every attribute [illegible] X of $\mathbf{R}$ an attribute of $\mathbf{R}$ not in X is functionally dependent on X only if X is a key of $\mathbf{R}$.

Unfortunately, this definition is again based on one relation scheme only. As a result, [illegible] element in the relational schema given in Example 1 above is in Boyce-Codd normal [illegible] Codd third normal form, yet the set suffers from unnecessary redundancy.

[illegible] drawback of the Boyce-Codd version of normalization is that given a [illegible] A [illegible] a set of functional dependencies F, there may not exist a relational schema [illegible] Codd normal form that covers F [Osborn 78]. Thus the definition does not [illegible] simultaneous attainment of all three goals: reconstructibility, covering, and [illegible].

---

† [illegible] are [illegible] sets {A,B,C,...} [illegible]

4. An improved third normal form

Because normalization should always be achieved in addition to covering and reconstructibility, the properties of a normal form will henceforth be discussed solely in the context of a relational schema that has the other two properties. Thus given A, a set of attributes, and F, a set of functional dependencies involving subsets of A, it is first desirable to obtain a relational schema that is satisfactory except for normalization. The following algorithm is based upon Bernstein's synthesis algorithm [Bernstein 76, p. 293]:

Preparatory algorithm.

Input. A, a set of attributes, and F, a set of functional dependencies on A.

1. (Remove extraneous attributes and dependencies.)
Eliminate from both sides of each functional dependency in F all attributes whose elimination leaves a set of functional dependencies having a closure equal to F^+. Next eliminate from that modified set all functional dependencies whose right side is the empty set of attributes. Let F_1 be the resulting set.

2. (Partition the functional dependencies.)
Partition F_1 into a set of classes C such that all the functional dependencies in each class have properly equivalent left sides; that is $V_1 \rightarrow W_1$ and $V_2 \rightarrow W_2$ are in the same class if and only if $V_1 \rightarrow V_2 \in F^+$ and $V_2 \rightarrow V_1 \in F^+$.

3. (Construct relation schemes.)
For each class in C, construct a relation scheme $\mathbf{R}_i$ consisting of all attributes A_i appearing in that class. Let $\mathbf{R}$ be the set resulting from these constructions.

4. (Augment the relational schema, if necessary.)
If for each relation scheme in $\mathbf{R}$, the set of attributes A_i in that class is such that $A_i \rightarrow A \notin F^+$, then construct A' to be a minimal set of attributes in A such that $A' \rightarrow A \in F^+$ and augment $\mathbf{R}$ by one relation scheme whose attribute set is A'.

Output. $\mathbf{R}$, the preparatory relational schema.

It is simple to show that the set of attributes constituting the left side of a functional dependency in F_1 is a key of the relation scheme having that dependency and constructed in Step 3; Bernstein has called each such set of attributes a *synthesized key*. Furthermore, if an additional relation scheme is introduced into $\mathbf{R}$ in Step 4, its synthesized key consists of all its attributes. To represent the dependencies contained in F, the set K_i of synthesized keys will be recorded as part of the relation schemes in $\mathbf{R}$. In the rest of this paper, the phrase "Let $\mathbf{R}$ be a preparatory relational schema ..." is shorthand for "Given a set of attributes A and a set of functional dependencies F defined on subsets of A, let $\mathbf{R}$ be a preparatory relational schema consisting of relation schemes $\mathbf{R}_i$, each having a set of attributes A_i and a set K_i of synthesized keys ..."; the notation used in examples will be $\mathbf{R}=\{\mathbf{R}_1<A_1,K_1>, \ldots, \mathbf{R}_n<A_n,K_n>\}$.

Let G_i be the set of synthesized functional dependencies in the relation scheme $\mathbf{R}_i$, that is, for each $i \geqslant 1$, $G_i=\{K \rightarrow A_i-K \mid K \in K_i\}$. For $G=\bigcup G_i$, $G^+=F^+$ [Bernstein 76]; that is, G covers the given functional dependencies, as described in Section 3. Osborn has proven that in the presence of [illegible], $\mathbf{R}$ has the desired property of reconstructibility if one of the relation schemes in $\mathbf{R}$ [illegible] key K such that $K \rightarrow A \in F^+$ [Osborn 77, Biskup 79]; this is guaranteed in Step 4. Thus [illegible] would classify the elements of $\mathbf{R}$ as independent components [Rissanen 77], and Beeri et al. would classify $\mathbf{R}$ as a "Rep4-representation" for A and F [Beeri 78].

Beeri and Bernstein have shown that Steps 1-3 can be computed in time proportional to the [illegible] of the length of the input [Beeri 79]. Since the last step is similar to repeating Steps 1-3 [illegible] bound; thus the preparatory algorithm runs in time $O(|F|^2|A|^2)$.

Normalization has been purposely omitted from this algorithm [illegible] the algorithm [illegible] Bernstein's [illegible], for which it is shown that the relation schemes [illegible] $\mathbf{R}$ [illegible] third normal form [Bernstein 76]. Given [illegible] relational schema [illegible] the preparatory algorithm for given sets of attributes and functional dependencies, the normalization procedure proposed here will remove attributes from individual schemes and adjust the set of synthesized keys. For simplicity, any such derived relational schema will also be called a preparatory relational schema as long as it maintains the properties of covering and reconstructibility.

The object of normalization is to remove unnecessary redundancy from a collection of relations. In particular, with respect to a relational schema **R**, an attribute B is *superfluous* in a relation scheme $\mathbf{R}_i$ if its removal from $\mathbf{R}_i$ does not affect covering nor reconstructibility; that is, all data relationships stored in an instance of **R** can be reconstructed without reference to the attribute B in $\mathbf{R}_i$. A more precise definition is given below.

Let **R** be a preparatory relational schema including $\mathbf{R}_i$, and let B be an attribute in A_i. The functional dependencies that do not involve B in $\mathbf{R}_i$ may be defined as follows:

$$D_i(\mathrm{B})=\bigcup_{j\neq i}\{\mathrm{X}\rightarrow A_j-\mathrm{X}\mid \mathrm{X}\subseteq A_j,\ \mathrm{X}\rightarrow A_j\in F^+,\text{ and for no }\mathrm{X}'\subset\mathrm{X},\ \mathrm{X}'\rightarrow A_j\in F^+\}$$

$$\cup\ \{\mathrm{X}\rightarrow A_i-\mathrm{X}-\mathrm{B}\mid \mathrm{B}\notin\mathrm{X},\ \mathrm{X}\subseteq A_i,\ \mathrm{X}\rightarrow A_i\in F^+,\text{ and for no }\mathrm{X}'\subset\mathrm{X},\ \mathrm{X}'\rightarrow A_i\in F^+\}$$

It is important to realize that $D_i(\mathrm{B})$ is defined in terms of all keys for all relation schemes in **R** (denoted by the union of terms), not only keys synthesized by the preparatory algorithm. Thus B is superfluous in $\mathbf{R}_i$ if both of the following conditions hold:

1. (covering condition): the set of dependencies excluding those involving B in $\mathbf{R}_i$ covers F; that is $D_i(\mathrm{B})^+=F^+$
2. (reconstructibility condition): a key of $\mathbf{R}_0$ (see Section 3) is contained in some relation without involving B in $\mathbf{R}_i$; that is, $A_j\rightarrow A\in F^+$ for some $j\neq i$ or $A_i-\mathrm{B}\rightarrow A\in F^+$.

Any algorithm that detects superfluous attributes by applying a straightforward implementation of these conditions requires the calculation of $D_i(\mathrm{B})^+$ for each possible value of i and B, which in turn requires that all keys of all relations be found. Because the number of keys can be exponential in $|A|$ and $|F|$ [Yu 76, Demetrovics 78], an algorithm used in practice must avoid calculating all keys. Thus rather than implementing the conditions as above, alternative definitions, less intuitive but more practical, will be given first.

Since the preparatory algorithm synthesizes only a polynomial number of keys (in terms of $|A|$ and $|F|$), it would be convenient to be able to ignore all keys that are not synthesized. As a parallel to $D_i(\mathrm{B})$, let $G_i'(\mathrm{B})$ be the set of all *synthesized dependencies that do not involve* B in $\mathbf{R}_i$; that is,

$$G_i'(\mathrm{B})=\bigcup_{j\neq i}G_j\ \cup\ \{\mathrm{K}\rightarrow A_i-\mathrm{K}-\mathrm{B}\mid \mathrm{B}\notin\mathrm{K}\text{ and }\mathrm{K}\in K_i\}.$$

An example will illustrate the difference between $D_i(\mathrm{B})$ and $G_i'(\mathrm{B})$:

EXAMPLE 3. Let A=ABCDE and F={A→B, B→A, AC→DE, BD→C}. For the preparatory relational schema **R**={$\mathbf{R}_1$<AB,{A,B}>,$\mathbf{R}_2$<ABCDE,{AC,BD}>}, it can be seen that $D_2(\mathrm{B})$={A→B, B→A, AC→DE, AD→CE} and $G_2'(\mathrm{B})$={A→B, B→A, AC→DE}. Notice that BD→ABCDE is in $D_2(\mathrm{B})^+$ but not in $G_2'(\mathrm{B})^+$; without the recognition of AD as a (non-synthesized) key of $\mathbf{R}_2$, B would not be seen to be superfluous in $\mathbf{R}_2$.

[illegible] R [illegible] R [illegible]

[illegible] R

1. B is *restorable* in $\mathbf{R}_i$ if $K_i\neq\{A_i\}$, i.e., A_i is not the only key of $\mathbf{R}_i$, and for every key K∈K_i such that B∉K, K→B∈$G_i'(\mathrm{B})^+$.
2. B is *non-essential* in $\mathbf{R}_i$ if for every key K∈K_i such that B∈K, there is a set K′ of attributes such that K′⊆A_i−B, K′→A_i∈G^+, and K→K′∈$G_i'(\mathrm{B})^+$, i.e., the closure of K relative to [illegible] (possibly non-synthesized) key K′ for A_i such that B∉K′.

$\mathbf{R}$. (An attribute that does not meet the conditions for the second definition is said to be *essential.*) Thus a non-prime attribute is obviously non-essential, and a transitive dependency will be shown to imply an attribute's restorability; these facts which will be used in the proof of Theorem 4. Together, the definitions characterize a superfluous attribute, as follows:

THEOREM 3. Let $\mathbf{R}$ be a preparatory relational schema including $\mathbf{R}_i$, and let B be an attribute in A_i. The attribute B is superfluous in $\mathbf{R}_i$ if and only if it is restorable and non-essential in $\mathbf{R}_i$.

> **PROOF:** Assume first that B is superfluous in $\mathbf{R}_i$. If $K_i=\{A_i\}$, then the relation $\mathbf{R}_i$ was introduced in Step 4 of the preparatory algorithm, and thus the reconstructibility condition for superfluous would be violated. Therefore, it must be true that $K_i \neq \{A_i\}$. Consider any synthesized key $K \in K_i$ such that $B \notin K$. Since $K \to B \in F^+$ and $F^+ = D_i(B)^+$ by hypothesis, $K \to B \in D_i(B)^+$, and thus by Lemma 3.1 below, B is restorable in $\mathbf{R}_i$. Now consider any synthesized key $K \in K_i$ such that $B \in K$. Since $K \to A_i \in F^+$ and $F^+ = D_i(B)^+$ by hypothesis, $K \to A_i \in D_i(B)^+$ and thus by Lemma 3.2 below, B is non-essential in $\mathbf{R}_i$.
>
> Next assume that B is restorable and non-essential in $\mathbf{R}_i$. Since $D_i(B)^+ \subseteq F^+ = G^+$, the proof of the second condition for superfluous requires that $G \subseteq D_i(B)^+$. Let g be an element of G, that is, g: $K \to A_j - K$ for some j.
>
> Case 1. $j \neq i$. By the definition of $D_i(B)$, $g \in D_i(B)$ and thus $g \in D_i(B)^+$.
>
> Case 2. $j=i$ and $B \in K$. If $K \to A_i \in G_i'(B)^+$ then $g \in D_i(B)^+$. Otherwise, since B is non-essential there is a key $K' \subseteq A_i$ such that $B \notin K'$, $K \to K' \in G_i'(B)^+$ and $K' \to A_i \in F^+$. Thus $K \to K' \in D_i(B)^+$ and $K \to A_i - B \in D_i(B)^+$ and thus that $g \in D_i(B)^+$.
>
> Case 3. $j=i$ and $B \notin K$. Since B is restorable in $\mathbf{R}_i$, $K \to B \in G_i'(B)^+$. Thus $g \in G_i'(B)^+$ and therefore $g \in D_i(B)^+$.
>
> It remains to be proven that [illegible] of $\mathbf{R}_0$ is contained in some relation without involving B in $\mathbf{R}$. If $A_j \to A \in F^+$ for $j \neq i$ then $A_j \to A \in F^+$ since $\mathbf{R}$ has the property of reconstructibility. In this case, it must be shown that $A_i - B \to A \in F^+$. If there is a key $K \in K_i$ such that $B \notin K$, then $K \to A \in F^+$, and thus $A_i - B \to A \in F^+$. Otherwise, since B is non-essential in $\mathbf{R}_i$, there is a non-synthesized key K for A_i such that $B \notin K$; thus again $A_i - B \to A \in F^+$. This completes the proof of the theorem.
>
> **LEMMA 3.1.** Let $\mathbf{R}$ be a preparatory relational schema including $\mathbf{R}_i$, and let B be an attribute in A_i. If $K_i \neq \{A_i\}$ and for every $K \in K_i$ such that $B \notin K$, $K \to B \in D_i(B)^+$, then B is restorable in $\mathbf{R}_i$.
>
> **PROOF:** Assume that for some $K \in K_i$ such that $B \notin K$, $K \to B \in D_i(B)^+$ but $K \to B \notin G_i'(B)^+$. Let $<X_0, X_1, \ldots, X_n>$ be a maximal derivation from K using $G_i'(B)$. By assumption, $B \notin X_n$. However $A_i - B \subseteq X_n$ since $K \to A_i - K - B \in G_i'(B)$. By hypothesis, there is a derivation $<X_n, X_{n+1}, X_{n+2}, \ldots, X_m>$ of B from X_n using $D_i(B)$. Since $A_i \not\subseteq X_n$, $m \geqslant n+1$ and thus there is a functional dependency f: $V \to W \in D_i(B) - G_i'(B)$, such that $V \subseteq X_n$, and $W \not\subseteq X_n$. Because $f \in D_i(B)$ it follows that for some j, $V \subseteq A_j$, $W \subseteq A_j - V$, and $V \to A_j \in F^+$; therefore $V \to A_j \in G^+$, where G is the set of all synthesized functional dependencies. Let $<X_0', X_1', \ldots, X_p'>$ be a derivation of W from V using G. Since $G_i'(B) \subseteq G$ and $V \to W \notin G_i'(B)$, this derivation must use a functional dependency in $G - G_i'(B)$; let $V_1 \to W_1$ be the first such dependency used in this derivation. From the definitions of G and $G_i'(B)$, it follows that V_1 is a synthesized key of $\mathbf{R}_i$. Thus $<X_0, X_1, \ldots, X_n>$ demonstrates that $K \to V \in G_i'(B)^+ \subseteq F^+$ and $<X_0', X_1', \ldots, X_p'>$ demonstrates that $V \to V_1 \in F^+$; therefore V, a key of $\mathbf{R}_j$, is functionally equivalent to some key of $\mathbf{R}_i$ and as a result $j=i$. However $A_i - B \subseteq X_n$, $W \not\subseteq X_n$, and $V \to W \in D_i(B)$ imply that $j \neq i$. This contradiction requires abandonment of the hypothesis, and therefore B is

restorable in $\mathbf{R}_i$.

Lemma 3.2. Let $\mathbf{R}$ be a preparatory relational schema including $\mathbf{R}_i$, and let B be an attribute in A_i. If for every key $K \in K_i$ such that $B \in K$, $K \to A_i \in D_i(B)^+$, then B is non-essential in $\mathbf{R}_i$.

Proof: Assume that for some $K \in K_i$ such that $B \in K$, $K \to A_i \in D_i(B)^+$. Let $<X_0, X_1, \ldots, X_n>$ be a maximal derivation from K using $G_i'(B)$. If $A_i \subseteq X_n$, then the following argument shows that there must be a key for A_i in $X_n - B$. If all keys for A_i contain B, then, by definition, $G_i'(B) = G - G_i$. Thus $K \to A_i \in (G - G_i)^+$, which can only result from a given functional dependency $f: K \to Z$ being redundant, that is, $f \in (F - f)^+$. However, each such functional dependency would have been removed in Step 1 of the preparatory algorithm and $\mathbf{R}_i$ would not have been created, thus proving that if $A_i \subseteq X_n$, then B is non-essential in $\mathbf{R}_i$.

If $A_i \not\subseteq X_n$, then let $<X_n, X_{n+1}, \ldots, X_m>$ be a derivation of A_i from X_n using $D_i(B)$, and let V and W be as defined in the proof of Lemma 3.1. Since, by the same argument, $j = i$ and thus V is a key of $\mathbf{R}_i$ and since $V \to W \in D_i(B)$ implies that $B \notin V$, the closure of K relative to $G_i'(B)$ has been shown to contain a key for A_i which does not contain B, and thus B is non-essential in $\mathbf{R}_i$.

Finally, using the definitions of restorable and non-essential, a characterization for normalization can be defined in a manner similar to the statement of Theorem 1 for Codd third normal form:

A relation scheme $\mathbf{R}_i$ in a preparatory relational schema $\mathbf{R}$ is in *improved third normal form* if each non-essential attribute is not restorable in $\mathbf{R}_i$.

Restorability in $\mathbf{R}_i$ indicates a form of "implicit" or "indirect" dependency on an arbitrarily chosen key that does not contain B [*cf.* Ling 78, Maier 79]; thus this definition is an exact analog for Theorem 1.

In re-examining the examples given in Section 3, it can be seen that the definition of improved third normal form captures the desired notion of non-redundancy.

Example 1′. Let A=ABCDEF and $F = \{AB \to CD, A \to E, B \to F, EF \to C\}$. The preparatory algorithm will yield the relational schema $\mathbf{R} = \{\mathbf{R}_1 <ABD, \{AB\}>, \mathbf{R}_2 <AE, \{A\}>, \mathbf{R}_3 <BF, \{B\}>, \mathbf{R}_4 <EFC, \{EF\}>\}$, since C is an extraneous attribute in $AB \to CD$. It can be seen that there are no attributes that are both non-essential and restorable in any of the relation schemes, which are therefore all in improved third normal form.

Example 2′. Let A=ABCDEF and $F = \{AD \to B, B \to C, C \to D, AB \to E, AC \to F\}$. For the relational schema $\mathbf{R} = \{\mathbf{R}_1 <ABCDEF, \{AB, AC, AD\}>, \mathbf{R}_2 <BC, \{B\}>, \mathbf{R}_3 <CD, \{C\}>\}$, it can be seen that the attribute C is non-essential and restorable in $\mathbf{R}_1$: using $G_1'(C) = \{B \to C, C \to D, AB \to DEF, AD \to BEF\}$, ABCDEF is derivable from the only synthesized key involving C (i.e., AC) and C is derivable from AB, a key not containing C. Removing C from $\mathbf{R}_1$ leaves a relational schema each member of which is in improved third normal form.

Theorem 4. Let $\mathbf{R}$ be a preparatory relational schema including $\mathbf{R}_i$. If $\mathbf{R}_i$ is in improved third normal form, then it is also in Codd third normal form.

Proof: The following lemma shows that transitive dependencies for non-prime attributes result in those attributes being restorable. Together with the observation that all non-prime attributes are non-essential (since a non-prime attribute B is in no $K \in K_i$), the conditions for Theorem 1 necessarily occur in improved third normal form schemes, thus proving this theorem.

Lemma 4.1. Let $\mathbf{R}$ be a preparatory relational schema including $\mathbf{R}_i$, and let B be a non-prime attribute in A_i. If B is transitively dependent on K, a key of $\mathbf{R}_i$, it is restorable in $\mathbf{R}_i$.

PROOF: If B is transitively dependent on K, then for some X contained in A_i, $B \notin X$, $K \rightarrow X \in F^+$, $X \rightarrow K \notin F^+$, and $X \rightarrow B \in F^+$. Since B is non-prime, $B \notin K$ and therefore clearly $K_i \neq \{A_i\}$. Since $B \notin K \cup X$, $K \rightarrow X \in G_i'(B)$ and therefore $K \rightarrow X \in G_i'(B)^+$. Because $X \rightarrow K \notin F^+$ and $X \rightarrow B \in F^+$, B is derivable from X using F whereas K is not. Thus, because the left side of each functional dependency used in the derivation $X \rightarrow B$ cannot be properly equivalent to K, each such dependency is placed by the preparatory algorithm in some class distinct from the one resulting in the construction of $\mathbf{R}_i$. Therefore B is derivable from X using $G - G_i \subseteq G_i'(B)$; that is, $X \rightarrow B \in G_i'(B)^+$. Hence, by transitivity, $K \rightarrow B \in G_i'(B)^+$. For each key K' in K_i such that $B \notin K'$, $K' \rightarrow K \in G_i'(B)$; thus $K' \rightarrow B \in G_i'(B)^+$ Therefore B is restorable in $\mathbf{R}_i$.

It should be noted that Theorem 4 does not claim a necessary, but rather a sufficient, condition for Codd third normal form. Together with Theorems 3 and 4, the examples show that improved third normal form is superior to Codd third normal form in removing superfluous attributes.

An efficient *deletion normalization algorithm* can be derived by starting with the preparatory relational schema and repeatedly removing superfluous attributes. Since the result of each removal is again a preparatory relational schema, eventually such a normalization algorithm will result in a preparatory set in which there are no superfluous attributes; that is, the result will be a relational schema in improved third normal form.

Before describing this algorithm formally, it is convenient to give a formal description of an algorithm that indicates whether or not a given attribute is superfluous in a relation scheme. The algorithm determines the restorability of an attribute by appealing to the following lemma:

LEMMA. Let $\mathbf{R}$ be a preparatory relational schema including $\mathbf{R}_i$, and let B be an attribute in A_i and K be a synthesized key with $B \notin K$. If $K \rightarrow B \in G_i'(B)^+$, then for each key K' in K_i such that $B \notin K'$, $K' \rightarrow B \in G_i'(B)^+$.

PROOF: If K' is a synthesized key and $B \notin K'$, then by definition $K' \rightarrow A_i - K' - B \in G_i'(B)$. Thus, since $B \notin K$, $K' \rightarrow K \in G_i'(B)^+$. It is given that $K \rightarrow B \in G_i'(B)^+$, and therefore $K' \rightarrow B \in G_i'(B)^+$.

Superfluous attribute detection algorithm

Input. $\mathbf{R}$, a preparatory relational schema; i, the index of some scheme in $\mathbf{R}$; B an attribute in A_i.

1. If $K_i = \{A_i\}$
then mark B non-superfluous and return
else mark B superfluous,

1.1. construct $K_i' = \{K \in K_i \mid B \notin K\}$

1.2. construct $G_i'(B)$ by (temporarily) removing all dependencies involving B in $\mathbf{R}_i$ from G

2. (Check restorability.)
If K_i' is not empty
then choose any key K from K_i'

2.1. If $K \rightarrow B \notin G_i'(B)^+$
then mark B non-superfluous and return

3. (Check non-essentiality.)
For each key K in $K_i - K_i'$ and while B is marked superfluous do

3.1. If $K \rightarrow A_i \notin G_i'(B)^+$

3.1.1. then let M denote the closure of K relative to $G_i'(B)$

3.1.2. if $(M \cap A_i) - B \rightarrow A_i \notin G^+$

then mark B non-superfluous

3.1.2.1. else insert into K_i' any key of $\mathbf{R}_i$ contained in $(M \cap A_i) - B$

Output. K_i' if B is marked superfluous and $\varnothing$ if B is marked non-superfluous.

Each substep for Step 3.1 runs in time $O(|G||A_i|)$ and is executed at most $O(|K_i|)$ times. Since Step 2.1 runs in time of the same order and is executed at most once, Steps 2 and 3 together require $O(|K_i||F||A|)$ time (at most one dependency in G results from each given functional dependency). Because Step 1.1 takes time $O(|K_i|)$ and Step 1.2 takes time $O(|F|)$, a single attribute can be checked in $O(|K_i||F||A|)$ time.

THEOREM 5. Let $\mathbf{R}$ be a preparatory relational schema including $\mathbf{R}_i$, and let B be an attribute in A_i. If B is not superfluous in $\mathbf{R}_i$, then it will not be superfluous in *any* relation scheme derived from $\mathbf{R}_i$ by the removal of superfluous attributes from a scheme in $\mathbf{R}$.

PROOF: The details of this proof are too lengthy to include here. The following statements highlight the arguments:

1. For two attributes B_1 and B_2 in $\mathbf{R}_i$, the part of $D_i(B_1)$ that does not involve B_2 in $\mathbf{R}_i$ is identical to the part of $D_i(B_2)$ that does not involve B_1 in $\mathbf{R}_i$. Furthermore, the closure of that part is contained in both $D_i(B_1)^+$ and $D_i(B_2)^+$.
2. Because of 1, an attribute that is not restorable in $\mathbf{R}_i$ cannot become restorable through the removal of other attributes in A_i.
3. Let B_1 be an attribute that is essential in $\mathbf{R}_i$ and K_i' be that set of keys from which A_i cannot be derived using $D_i(B_1)$. If another attribute B_2 is in all keys in K_i', then B_2 is also essential.
4. From 1 and 3, an attribute that is essential in $\mathbf{R}_i$ cannot become non-essential through the removal of other non-essential attributes in A_i.
5. Removal of attributes from other schemes in $\mathbf{R}$ does not affect whether or not B is superfluous in $\mathbf{R}_i$.

The result of Theorem 5 is that each attribute in $\mathbf{R}$ must be tested once only: once found to be non-superfluous it need not be re-examined after removing other attributes. Hence the complete normalization algorithm is as follows:

Deletion normalization algorithm.

Input. A, a set of attributes; F, a set of functional dependencies on A.

1. (Prepare a relational schema.)
Use the preparatory algorithm for A and F to yield $\mathbf{R}$.

2. (Test each relation scheme for superfluous attributes.)
For i := 1 to $|\mathbf{R}|$ do

2.1. (Test each attribute in A_i.)
For each B in A_i do

If the superfluous attribute detection algorithm returns a non-[illegible]
for $\mathbf{R}$, i, and B then

2.1.1. Construct $\mathbf{R}_i'$ such that $A_i' = A_i - B$ and K_i' is the returned [illegible]

2.1.2. Replace $\mathbf{R}_i$ by $\mathbf{R}_i'$ in $\mathbf{R}$.

Output. $\mathbf{R}$, a relational schema in improved third normal form.

Given a particular relation scheme $\mathbf{R}_i$, because in Step 2.1 the superfluous attribute detection algorithm is called for each attribute in A_i, the time for that step is bounded by $O(|K_i||F||A|^2)$. (In Step 2.1.1, the size of K_i' is always less than or equal to the size of K_i since the algorithm introduces at most one new key for each key removed by the elimination of B, as implied by the proof for Theorem 3.) Step 2.1, in turn, is repeated for each relation scheme in $\mathbf{R}$, where each functional dependency results in the appearance of at most one key in some one scheme in $\mathbf{R}$. Since the number of keys in total is therefore bounded by $|F|+1$ (the extra key resulting from Step 4 of the preparatory algorithm where a relation scheme may be inserted into $\mathbf{R}$ for reconstructibility), Step 2 takes time $O(|F|^2|A|^2)$. Because Step 1 also requires time of the same order, that is the bound for the complete algorithm.

5. Conclusions

We have shown that some Codd third normal form relation schemes and even some Boyce-Codd normal form schemes still contain simply removable superfluous attributes because the definitions of transitive dependency and non-prime attribute are inadequate when applied to sets of schemes. We defined restorable and non-essential to replace those definitions and proved that in a preparatory relational schema, a transitive dependency implies the presence of a restorable attribute and an attribute that is essential is always prime. We were then able to define an improved third normal form which is superior to Codd third normal form in removing superfluous attributes. Furthermore we have proven that no superfluous attributes remain. We then presented the deletion normalization algorithm which produces a relational schema in improved third normal form while guaranteeing covering and reconstructibility. The complexity of the algorithm is $O(|F|^2|A|^2)$ which is the same as that of the best known algorithms for generating relation schemes in Codd third normal form.

It should be noted that the removal of all superfluous attributes does not necessarily imply the absence of update anomolies. In particular, if an attribute B in $\mathbf{R}_i$ satisfies the covering condition but not the reconstructibility condition, then although it is not superfluous, its updating may lead to anomolous behaviour.

It is interesting to contrast the deletion normalization method with the decomposition method for normalization [Codd 71, Delobel 73, Rissanen 77]. Decomposing a relation scheme $\mathbf{R}$ into two schemes $\mathbf{R}_1$ and $\mathbf{R}_2$ requires that $\mathbf{R}$ can be reconstructed from $\mathbf{R}_1$ and $\mathbf{R}_2$. Let A_1, A_2, and A be the sets of attributes for $\mathbf{R}_1$, $\mathbf{R}_2$, and $\mathbf{R}$, respectively, and let F be the set of functional dependencies for $\mathbf{R}$. The reconstructibility of $\mathbf{R}$ requires a *lossless join* of $\mathbf{R}_1$ and $\mathbf{R}_2$, which has been shown to occur if and only if either A_1 or A_2 is functionally dependent on their intersection, that is, the intersection contains a key of $\mathbf{R}_1$ or of $\mathbf{R}_2$ [Rissanen 77, Aho 79]. Without loss of generality, assume that $A_1 \cap A_2 \rightarrow A_2 \in F^+$. Thus $A_1 \rightarrow A_1 \cup A_2$ or $A_1 \rightarrow A \in F^+$, which implies that each key of $\mathbf{R}_1$ is also a key of $\mathbf{R}$. Since $A_1 \cap A_2 \rightarrow A_2 - A_1 (= A - A_1) \in F^+$, $X = A_1 \cap A_2$ is a set of attributes in $\mathbf{R}_1$ and $\mathbf{R}_2$ such that all those attributes which are in $\mathbf{R}$ (and $\mathbf{R}_2$) but not in $\mathbf{R}_1$ are functionally dependent on X in $\mathbf{R}$. If $\mathbf{R}_1$ and $\mathbf{R}_2$ are distinct relation schemes in a preparatory relational schema, then $X \rightarrow K \notin F^+$ where K is a key of $\mathbf{R}_1$ (otherwise the relations must be combined). When $\mathbf{R}$ is (non-trivially) decomposed into $\mathbf{R}_1$ and $\mathbf{R}_2$, there is at least one attribute B in $A_2 - A_1$. In this case, $K \rightarrow X \in F^+$, $X \rightarrow B \in F^+$, and $X \rightarrow K \notin F^+$, that is, $K \rightarrow B \in F^+$ is a transitive dependency in $\mathbf{R}$. Hence the decomposition method for normalizing a relation scheme is applicable if and only if there exists a transitive dependency within the relation scheme. Furthermore, if B is transitively dependent on some key in $\mathbf{R}$, the result of applying the deletion normalization algorithm to B is the same as the result of applying decomposition. Thus the deletion normalization method is more powerful than the decomposition method for normalization.

The definitions suggested here can easily be extended to give an improved version of Boyce-Codd normal form as follows:

A relation scheme $\mathbf{R}_i$ in a relational schema $\mathbf{R}$ is in *improved Boyce-Codd normal form* if no attribute is restorable in $\mathbf{R}_i$.

It can easily be shown that any relation scheme in improved Boyce-Codd normal form is also in both Boyce-Codd normal form and improved third normal form. It must be remembered, however, that for a given A and F, a covering Boyce-Codd normal form may not exist, and thus it is not necessarily possible to find a *preparatory* relational schema each element of which is in improved Boyce-Codd normal form.

Finally, throughout this paper the notion of dependency has been restricted to functional dependencies only. The presence of other forms of dependency, such as multivalued dependencies [Fagin 77b], first order hierarchical decompositions [Delobel 78], and other constraints among attribute values also give rise to potential redundancy among the relation schemes. Further research may be needed to establish adequate definitions and algorithms for the removal of those forms of redundancy as well.

[illegible] of this paper, and we give special thanks to David Maier and the referees.

References

Aho 79 — Aho, A.V., Beeri, C., and Ullman, J.D. The theory of joins in relational data bases. *ACM Trans. on Database Systems 4,* 3 (Sept. 1979) 297-314.

Armstrong 74 — Armstrong, A.A. Dependency structures of data base relationships. *IFIP 74,* North Holland (1974) 580-583.

Beeri 78 — Beeri, C., Bernstein, P.A., and Goodman, N. A sophisticate's introduction to database normalization theory. *Proc. of the Fourth Int. Conf. on Very Large Data Bases* (1978) 113-124.

Beeri 79 — Beeri, C. and Bernstein, P.A. Computational problems related to the design of normal form relational schemes. *ACM Trans. on Database Systems 4,* 1 (March 1979) 30-59.

Bernstein 76 — Bernstein, P.A. Synthesizing third normal form relations from functional dependencies. *ACM Trans. on Database Systems 1,* 4 (Dec. 1976) 277-298.

Biskup 79 — Biskup, J., Dayal, U., and Bernstein, P.A. Synthesizing independent database schemes. *Proc. of the Int. Conf. on Management of Data,* ACM Sigmod (1979) 143-151.

Codd 70 — Codd, E.F. A relational model for large shared data banks. *Comm. of ACM 13,* 6 (June 1970) 377-387.

Codd 71 — Codd, E.F. Further normalization of the relational data base model. Courant Computer Science Symp. 6, *Data Base Systems,* R. Rustin (ed.), Prentice-Hall (1971) 33-64.

Codd 74 — Codd, E.F. Recent investigations in relational data base systems. *IFIP 74,* North-Holland (1974) 1017-1021.

Delobel 73 — Delobel, C. and Casey, R.G. Decomposition of a data base and the theory of Boolean switching functions. *IBM J. of Res. and Dev. 17,* 5 (Sept. 1973) 374-386.

Delobel 78 — Delobel, C. Normalization and hierarchical dependencies in the relational data model. *ACM Trans. on Database Systems 3,* 3 (Sept. 1978) 201-222.

Demetrovics 78 — Demetrovics, J. On the number of candidate keys. *Inf. Proc. Letters 7,* 6 (1978) 266-269.

Fagin 77a	Fagin, R. Functional dependencies in a relational database and propositional logic. *IBM J. of Res. and Devel. 21*, 6 (Nov. 1977) 534-544.
Fagin 77b	Fagin, R. Multivalued dependencies and a new normal form for relational databases. *ACM Trans. on Database Systems 2*, 3 (Sept. 1977) 262-278.
Kent 73	Kent, W. A primer of normal forms. IBM Systems Development Division TR02.600 (Dec. 1973).
Ling 78	Ling, T.-W. Improving data base integrity based on functional dependencies. Ph.D. dissertation, Dept. of Computer Science, Univ. of Waterloo (1978).
Lucchesi 78	Lucchesi, C.L. and Osborn, S.L. Candidate keys for relations. *J. of Comp. and Sys. Sci. 17*, 2 (1978) 270-279.
Maier 79	Maier, D. Minimum covers in the relational database model. *Proc. of the 11th Ann. ACM Symp. on Theory of Computing.* (1979) 330-337.
Osborn 77	Osborn, S.L. Normal forms for relational data bases. Ph.D. dissertation, Dept. of Computer Science, Univ. of Waterloo (1977).
Osborn 78	Osborn, S.L. Testing for existence of a covering Boyce-Codd normal form. *Inf. Proc. Letters 8*, 1 (1978) 11-14.
Rissanen 77	Rissanen, J. Independent components of relations. *ACM Trans. on Database Systems 2*, 4 (Dec. 1977) 317-325.
Yu 76	Yu, C.T. and Johnson, D.T. On the complexity of finding the set of candidate keys for a given set of functional dependencies. *Inf. Proc. Letters 5*, 4 (1976) 100-101.