Direct Product Decompositions of Lattices, Closures and Relation Schemes

Leonid Libkin*

Department of Computer and Information Science
University of Pennsylvania
Philadelphia, PA 19104-6389, USA

Abstract

In this paper we study the direct product decompositions of closure operations and lattices of closed sets. We characterize the direct product decompositions of lattices of closed sets in terms of closure operations, and find those decompositions of lattices which correspond to the decompositions of closures. If a closure on a finite set is represented by its implication base (i.e. a binary relation on the powerset), we construct a polynomial algorithms to find its direct product decompositions. The main characterization theorem is also applied to define direct product decompositions of relational database schemes and to find out what properties of relational databases and schemes are preserved under the decompositions.

1 Introduction

In [DFK] Demetrovics, Füredi and Katona introduced the concept of direct product decomposition of a closure operation. If C_1 and C_2 are two closures on disjoint sets U_1, U_2, then the direct product $C_1 \times C_2$ is a closure on $U_1 \cup U_2$ defined by

$$C_1 \times C_2(X) = C_1(X \cap U_1) \cup C_2(X \cap U_2),\ X \subseteq U_1 \cup U_2.$$

*Research partially supported by NSF Grants IRI-86-10617 and CCR-90-57570 and ONR Grant NOOO14-88-k0634.

If L_1 and L_2 stand for the lattices of closed sets of C_1 and C_2 respectively, then the lattice of closed sets of $C_1 \times C_2$ is the direct product $L_1 \times L_2$. However, it is unclear if every direct product decomposition of a lattice of closed sets corresponds to a direct product decomposition of the underlying closure in the sense of the operation $\times$ defined above. In the other words, if L_C is the lattice of closed sets of C and L_C is isomorphic to the direct product, $L_C \simeq \mathcal{L}_1 \times \mathcal{L}_2$, does it mean that $\mathcal{L}_1 \simeq L_{C_1}$ and $\mathcal{L}_2 \simeq L_{C_2}$, where $C = C_1 \times C_2$?

We are going to show in this paper that, generally speaking, the answer is "no". We do that by finding a characterization of the direct product decompositions of a lattice of closed sets in terms of the closure operation in section 2. This characterization will emphasize the importance of the operation $\times$. We will show that every lattice of closed sets of a closure C is isomorphic to the lattice of closed sets of a closure C' such that the direct product decompositions of this lattice are in 1-to-1 correspondence with the direct product decompositions of C'.

In the finite case, a closure on a set U can be represented by its *implication bases* [Wi] which consist of expressions of form $X \to Y$, $X, Y \subseteq U$. (E.g., we can represent a closure C by $\{X \to Y : Y \subseteq C(X)\}$). In section 3 we give some necessary facts about implication bases and then construct an algorithm finding the direct product decompositions of the closure represented by an implication base. This algorithm allows us to construct a direct product decomposition of a closure in polynomial time in the size of input, i.e. the implication base.

In short section 4 we show that our main characterization can be applied to obtain results describing the direct decompositions of some known classes of lattices and closures.

When speaking about relational databases, implication systems correspond exactly to *relation schemes*. A relation scheme is a pair $\langle U, F\rangle$ consisting of a set U and a family F of *functional dependencies*, the last being a set of expressions of form $X \to Y$, $X, Y \subseteq U$. We study the direct product decompositions of relation schemes in section 5. This is also of practical importance, because, as we will see, these direct product decompositions can describe decompositions of a relation scheme into several relation schemes within one database scheme and some nice properties, as being in a normal form, are preserved under decompositions. By the results of section 3, these direct product decompositions can be found in a polynomial time.

Now we introduce some terminology.

Throughout the paper, C (possibly, with indices) will denote a *closure operation* (or simply *closure*) on a set U, i.e. C is a map $C : \mathbf{P}(U) \to \mathbf{P}(U)$ such that
(C1) $\forall X \subseteq U : X \subseteq C(X)$;
(C2) $\forall X \subseteq Y \subseteq U : C(X) \subseteq C(Y)$;
(C3) $\forall X \subseteq U : C(C(X)) = C(X)$.

A set $X \subseteq U$ is called *closed* (w.r.t. C) if $C(X) = X$. Denote the family of all closed sets by L_C. Then L_C equipped with the natural ordering is a *lattice* in which *sup* and *inf* operations are defined by

$$\forall X, Y \in L_C : X \wedge Y = X \cap Y; X \vee Y = C(X \cup Y).$$

L_C thus constructed is a *complete* lattice [Bi].

We will always suppose that a closure C satisfies
(C4) $C(\emptyset) = \emptyset$.
Really, if $C(\emptyset) = X \neq \emptyset$, define $C'(Y) = C(Y) - X$ for $Y \subseteq U - X$. Then C' is a closure on $U - X$ satisfying (C4), and the lattices L_C and $L_{C'}$ are isomorphic. Hence, (C4) will not lead us to the loss of generality.

When speaking about an arbitrary lattice (not necessarily lattice of closed sets), we denote it by $\mathcal{L}$ and its elements by small letters.

If $\mathcal{L}$ is a finite lattice [1], there is a simple way to construct a closure C on a finite set U such that $\mathcal{L} \simeq L_C$, where $\simeq$ stands for the isomorphism. Let U be the set of *join-irreducible* elements $J(\mathcal{L})$, i.e. $U = \{a \in \mathcal{L} : (a = x \vee y) \Rightarrow (a = x \text{ or } a = y)\}$. Given $X \subseteq U$, let $C(X) = \{x \in U : x \leq \bigvee X\}$. Then C is a closure on U, and $L_C \simeq \mathcal{L}$.

If $\mathcal{L}$ is a *bounded* lattice, i.e. it contains the greatest element $\mathbf{1}$ and the least element $\mathbf{0}$, then $\overline{a}$ stands for a complement of a if it exists, that is, $a \wedge \overline{a} = \mathbf{0}$ and $a \vee \overline{a} = \mathbf{1}$.

We will need the concept of a *neutral* element. An element $a \in \mathcal{L}$ is called neutral [Bi],[Gr] iff for every $x, y \in \mathcal{L}$ the following holds

$$(a \vee x) \wedge (a \vee y) \wedge (x \vee y) = (a \wedge x) \vee (a \wedge y) \vee (x \wedge y).$$

In sequel we will use a more convenient form of this definition. An element $a \in \mathcal{L}$ is neutral iff for every $x, y \in \mathcal{L}$ the sublattice $\langle a, x, y \rangle$ generated by a, x, y is distributive [Gr].

If L_C is the lattice of closed sets of a closure C, and $A \in L_C$, then $(A]$ is the *principal ideal of* L_C generated by A, i.e. $(A] = \{X \in L_C : X \subseteq A\}$. In an arbitrary lattice, $(a]$ and $[a)$ stand for the principal ideal and coideal (filter) generated by a.

2 Direct product decompositions of lattices and closures

In this section we are going to answer two questions. The first one is: given a closure C on U such that L_C is isomorphic to the direct product of two lattices, $L_C \simeq \mathcal{L}_1 \times \mathcal{L}_2$, what can be said about C? In the other words, what are necessary and sufficient conditions that provide L_C to be isomorphic to the direct product of two lattices? The second question is: what is the relationship between direct product decompositions of closures and of lattices of closed sets?

We will see soon that if $L_C \simeq \mathcal{L}_1 \times \mathcal{L}_2$ then both $\mathcal{L}_1$ and $\mathcal{L}_2$ are isomorphic to lattices of closed sets of closures defined on two disjoint subsets of U. This explains why we characterize only decompositions into products of two lattices.

[1] It is enough to require that the dual lattice $\mathcal{L}^*$ be Noetherian [Bi].

Our first result describes the direct product decompositions of form $L_C \simeq \mathcal{L}_1 \times \mathcal{L}_2$.

Theorem 1 *Every direct product decomposition $L_C \simeq \mathcal{L}_1 \times \mathcal{L}_2$ has form $L_C \simeq (A] \times (\overline{A}]$ where $A, \overline{A} \in L_C$, $\overline{A}$ is a complement of A in L_C and A is neutral.*

More precisely, $\mathcal{L}_1 \simeq (A]$ and $\mathcal{L}_2 \simeq (\overline{A}]$, or $\mathcal{L}_1 \simeq (\overline{A}]$ and $\mathcal{L}_2 \simeq (A]$. However, in this case we prefer to speak of the direct product decomposition having form $L_C \simeq (A] \times (\overline{A}]$.

Proof. First, notice that a neutral element $a \in \mathcal{L}$ may not have two complements. Really, if it has two complements, $\overline{a}$ and $\tilde{a}$, then the sublattice $\langle a, \overline{a}, \tilde{a} \rangle = \{a, \overline{a}, \tilde{a}, \mathbf{1}, \mathbf{O}\}$ is not distributive. Since L_C is a bounded lattice, the following lemma finishes the proof.

Lemma 1 *If L is a bounded lattice, each direct product decomposition $\mathcal{L} \simeq \mathcal{L}_1 \times \mathcal{L}_2$ has form $\mathcal{L} \simeq (a] \times (\overline{a}]$, where a is a neutral element and $\overline{a}$ its complement.*

Proof of lemma. It is well-known that each direct product decomposition has form $\mathcal{L} \simeq (a] \times [a)$ [Gr]. Hence, we only have to prove that if a is a neutral complemented element, then $[a) \simeq (\overline{a}]$.

Define $\phi : (\overline{a}] \to [a)$ as follows: $\phi(x) = x \vee a$. Let $x \geq a$. Then $\phi(x \wedge \overline{a}) = (x \wedge \overline{a}) \vee a = x$ since the sublattice generated by $a, \overline{a}, x$ is distributive. Further, for $x \leq \overline{a}$ we have $\phi(x) \wedge \overline{a} = (x \vee a) \wedge \overline{a} = x$, i.e. $x_1 \neq x_2$ implies $\phi(x_1) \neq \phi(x_2)$. Thus, ϕ is a bijection. It follows from the definition that $\phi(x \vee y) = \phi(x) \vee \phi(y)$, and from the distributivity of $\langle a, x, y \rangle$ that $\phi(x \wedge y) = (x \wedge y) \vee a = (x \vee a) \wedge (y \vee a) = \phi(x) \wedge \phi(y)$. Hence, ϕ is an isomorphism. Lemma and theorem 1 are proved. □

Since $\mathcal{L} \simeq (a] \times [a)$ holds for every neutral element $a \in \mathcal{L}$, we obtain from theorem 1 and the proof of lemma 1

Corollary 1 *Given a closure C on U, there is a one-to-one correspondence between the direct product decompositions $L_C \simeq \mathcal{L}_1 \times \mathcal{L}_2$ and pairs $(A, \overline{A})$, where A is a neutral complemented element of L_C and $\overline{A}$ its complement.* □

Corollary 2 *If A is a neutral complemented element of L_C, then so is its complement $\overline{A}$.* □

Now we can introduce our main definition to be studied in sequel.

Definition. Given a closure C on U, a pair $(A, \overline{A})$ consisting of a neutral complemented element of L_C and its complement is called a *decomposition pair* (of C or of L_C).

We have shown so far that there is a one-to-one correspondence between the decomposition pairs and the direct product decompositions of L_C having form $L_C \simeq \mathcal{L}_1 \times \mathcal{L}_2$. Our next theorem, which is the main result of this section, gives a characterization of the decomposition pairs of an arbitrary closure. However, before presenting this theorem, we mention that considering only direct product decompositions of form $L_C \simeq \mathcal{L}_1 \times \mathcal{L}_2$ does not cause the loss of generality. This is true in view of the following

Corollary 3 *Let $L_C \simeq \mathcal{L}_1 \times \mathcal{L}_2$. Then both $\mathcal{L}_1$ and $\mathcal{L}_2$ are the lattices of closed sets.*

Proof of corollary. According to theorem 1 and corollary 2, $\mathcal{L}_1 \simeq (A]$ and $\mathcal{L}_2 \simeq (\overline{A}]$ for a decomposition pair $(A, \overline{A})$. Hence, $\mathcal{L}_1 \simeq L_{C|A}$ and $\mathcal{L}_2 \simeq L_{C|\overline{A}}$. □

Now we can give a characterization of the decomposition pairs of a closure.

Theorem 2 *A pair $(A, \overline{A})$ of disjoint subsets of a set U is a decomposition pair of a closure C on U iff the following hold:*
(i) $\forall X \subseteq A \cup \overline{A} : C(X \cap A) = C(X) \cap A$;
(ii) $\forall X \subseteq A \cup \overline{A} : C(X \cap \overline{A}) = C(X) \cap \overline{A}$;
(iii) $\forall X \subseteq U : C(X) = C(C(X) \cap (A \cup \overline{A}))$.

Proof. We start with a simple lemma.

Lemma 2 *A pair $(A, \overline{A})$ of disjoint subsets of U is a decomposition pair of L_C iff $A \vee \overline{A} = U$ and $\phi : L_C \to (A] \times (\overline{A}]$ given by $\phi(X) = (X \cap A, X \cap \overline{A})$ is an isomorphism.*

Proof of lemma. Let ϕ thus constructed be an isomorphism. Then $\phi(A) = (A, \emptyset)$, and by [Gr, Th.3.2.4] A is a neutral element of L_C. Analogously, so is $\overline{A}$. Hence, $(A, \overline{A})$ is a decomposition pair. Conversely, if $(A, \overline{A})$ is a decomposition pair, consider a mapping $\psi : (A] \times (\overline{A}] \to L_C$ given by $\psi(X, Y) = X \vee Y$. According to the definition of a neutral element, $(X \cap A) \vee (X \cap \overline{A}) = X$ and for $X \subseteq A$, $Y \subseteq \overline{A}$: $(X \vee Y) \cap A = X, (X \vee Y) \cap \overline{A} = Y$, i.e. $\psi = \phi^{-1}$. It shows immediately that ϕ is a one-to-one correspondence. Obviously, ϕ preserves the ordering, i.e. if $X \subseteq Y$ then $\phi(X) \leq \phi(Y)$ in $(A] \times (\overline{A}]$, and so does ψ. Hence, ϕ is an isomorphism. Lemma is proved.

Now we return to the proof of theorem 2. Let $(A, \overline{A})$ be a decomposition pair of C. Consider arbitrary $X \subseteq U$ and $C(X)$. Since $(C(X) \wedge A) \vee (C(X) \wedge \overline{A}) = C(X)$ according to the proof of lemma 2, we have $C(X) = C((C(X) \cap A) \cup (C(X) \cap \overline{A})) = C(C(X) \cap (A \cup \overline{A}))$, i.e. (iii) holds.

Let $X \subseteq A \cup \overline{A}$, and $Y = X \cap A, Z = X \cap \overline{A}$. Then $X = Y \cup Z$, and $C(Y) \subseteq A$, $C(Z) \subseteq \overline{A}$. We have $C(X) = C(Y \cup Z) = C(C(Y) \cup C(Z)) = C(Y) \vee C(Z)$, and $C(X) \cap A = (C(Y) \vee C(Z)) \wedge A = C(Y)$ since $\phi \cdot \psi = id$. Hence, $C(X) \cap A = C(X \cap A)$, and (i) holds. Analogously we prove that (ii) holds.

Let, conversely, (i) (ii) (iii) hold. Prove that A and $\overline{A}$ are complemented elements, and that ϕ from lemma 2 is an isomorphism.

Since A and $\overline{A}$ are disjoint, $A \wedge \overline{A} = \emptyset$. If $X = U$, we get from (iii) that $C(A \cup \overline{A}) = U$, i.e. $A \vee \overline{A} = U$. Hence, $\overline{A}$ is a complement of A.

We prove now that ϕ is a bijection. To do this, we need to prove two claims. Recall that $\psi(X, Y) = X \vee Y$.

Claim 1. $\phi \cdot \psi = id$ (more precisely, $id_{(A] \times (\overline{A}]}$).
Let $C(Y) \in (A]$, $C(Z) \in (\overline{A}]$, $Y \subseteq A$, $Z \subseteq \overline{A}$. Then $\psi(C(Y), C(Z)) = C(Y) \vee C(Z) = C(Y \cup Z)$. The first component of $\phi \cdot \psi(C(Y), C(Z))$ is $(C(X) \vee C(Z)) \wedge A = C(Y \cup Z) \cap A =$ (by (i)) $= C((Y \cup Z) \cap A) = C(Y)$. Analogously, by (ii) the second component of $\phi \cdot \psi(C(Y), C(Z))$ is $C(Z)$. Hence, $\phi \cdot \psi = id$.

Claim 2. $\psi \cdot \phi = id$ (more precisely, id_{L_C}).
Let $C(X)$ be an arbitrary element of L_C. Then we have $\psi \cdot \phi(C(X)) = (C(X) \wedge A) \vee (C(X) \wedge \overline{A}) = C((C(X) \cap A) \cup (C(X) \cap \overline{A})) = C(C(X) \cap (A \cup \overline{A})) = C(X)$ by (iii). Hence, $\psi \cdot \phi = id$.

It follows from two proved claims that ϕ is a bijection. Hence, the following finishes the proof.

Claim 3. ϕ is a homomorphism.
Clearly, ϕ is a $\wedge$-homomorphism. Hence, we must prove that for arbitrary $C(X), C(Y) \in L_C$ it holds : $\phi(C(X) \vee C(Y)) = \phi(C(X)) \vee \phi(C(Y))$. According to (iii) we may assume without loss of generality that $Y, Z \subseteq A \cup \overline{A}$. Further, $(C(X) \vee C(Y)) \wedge A = C(C(X) \cup C(Y)) \cap A = C(X \cup Y) \cap A =$ (by (i)) $= C((X \cup Y) \cap A) = C((X \cap A) \cup (Y \cap A)) = C(C(X \cap A) \cup C(C(Y \cap A)) =$ (by (i)) $= C((C(X) \cap A) \cup (C(Y) \cap A)) = (C(X) \wedge A) \vee (C(Y) \wedge A)$. Analogously, $(C(X) \vee C(Y)) \wedge \overline{A} = (C(X) \wedge \overline{A}) \vee (C(Y) \wedge \overline{A})$. Hence, ϕ is a $\vee$ - homomorphism too.

Thus, ϕ is a one-to-one homomorpism, i.e. an isomorphism. According to lemma 2, $(A, \overline{A})$ is a decomposition pair. Theorem is completely proved. □

As a corollary of theorem 2 we obtain a characterization of the direct product decompositions of closures. Let us call a decomposition pair $(A, \overline{A})$ *strong* if it is a partition of U, i.e. $A \cup \overline{A} = U$.

Corollary 4 *A partition $(A, \overline{A})$ of a set U is a strong decomposition pair of a closure C on U iff $\forall X \subseteq U : C(X) = C(X \cap A) \cup C(X \cap \overline{A})$.* □

Therefore, there is a one-to-one correspondence between the direct product decompositions of closures as they were introduced in [DFK], and the strong decomposition pairs of lattices of closed sets. In particular, not every direct product decomposition of lattice of closed sets corresponds to a direct product decomposition of a closure, because there exist decomposition pairs with $A \cup \overline{A} \neq U$. However, in the finite case for every closure there exists an "equivalent" one (i.e. having isomorphic lattice of closed sets) whose decomposition pairs are strong.

Proposition 1 *For every finite lattice $\mathcal{L}$ there is a finite set U and a closure C on U such that $\mathcal{L} \simeq L_C$ and all the decomposition pairs of C are strong.*

Proof. Consider the representation with $U = J(\mathcal{L})$ and $C(X) = J(\mathcal{L}) \cap (\bigvee X]$, see introduction. $L_C \simeq \mathcal{L}$ for this representation. Let $(A, \overline{A})$ be a decomposition pair of C. Suppose for $x \in \mathcal{L}$: $J(x) = \{y \in J(\mathcal{L}) : y \leq x\}$. Then $J(x) \in L_C$ if $x \in J(\mathcal{L})$. According to (iii) $J(x) = C(J(x) \cap (A \cup \overline{A}))$, i.e. $x \leq \bigvee(y : y \leq x, y \in A \cup \overline{A}) \leq x$. Hence, $x = \bigvee(y : y \leq x, y \in A \cup \overline{A})$, and since $x \in J(\mathcal{L})$, $x = y$ for some y, i.e. $x \in A \cup \overline{A}$. Therefore, $(A, \overline{A})$ is strong. □

3 Implication bases of closures and direct product decompositions

The main aim of this section is to present an algorithm finding a strong decomposition pair, i.e. a direct product decomposition of a closure. To construct such an algorithm, we must have a representation of closures. The most convenient way to represent a closure is to represent it by its *implication base* [Wi]. We introduce the definition of implication bases of finite closures, and then give a polynomial algorithm that, given an implication base of a closure, finds a strong decomposition pair of this closure, i.e. its direct product decomposition.

Given a finite set U, an *implication system* is a family $F = \{X \to Y : X, Y \subseteq U\}$. If we are given an implication system F, construct a map $C_F : \mathbf{P}(U) \to \mathbf{P}(U)$ using the following algorithm.

Algorithm CLOSURE
Input: an implication system F over U and a set $X \subseteq U$.
Output: $C_F(X)$
Method:
$result := X$;
WHILE there exists $Z \to Y \in F$ such that
    $Z \subseteq result$ AND $Y \not\subseteq result$
DO $result := result \cup Y$ END;
RETURN($result$).

It is well-known (see [Ar],[DLM1],[DLM2],[Ma],[Wi]) that C_F is a closure and for every closure over U there is an implication system on U generating this closure. We will call F an *implication base* of a closure C if $C = C_F$.

If $X = \{x\}$ and $Y = \{y\}$, we will write $x \to y$ instead of $X \to Y$. We first investigate a particular case when all the implications from F have form $x \to y$. Later we will see that finding strong decomposition pairs for such implication bases is the crucial step in the general algorithm.

Implications $x \to y$ were called *unary* in [MR2]. A characterization of implication systems consisting of unary implications was given in [DLM2].

Proposition 2 [DLM2]. *Given a closure C on a finite set U, the following are equivalent:*
(i) *C has an implication base consisting of unary implications;*
(ii) *C is topological, i.e. $C(X \cup Y) = C(X) \cup C(Y)$;*
(iii) *L_C is a sublattice of $\langle \mathbf{P}(U), \cap, \cup \rangle$.* $\square$

Corollary 5 *If C is a topological closure on a set U, then $(A, \overline{A})$ is a strong decomposition pair iff both A and $\overline{A}$ are closed and $(A, \overline{A})$ is a partition of U.*

Proof follows immediately from theorem 2 and corollary 4. $\square$

Let F be an implication system over U consisting only of unary implications. Define a graph $G_F^0 = (U, V^0)$, where U is a set of vertices and V is a set of edges, $V = \{(x, y) : x \to y \in F$ or $y \to x \in F\}$. Let $G_F = (U, V)$ be an undirected graph which is the symmetric transitive closure of G_F^0.

Proposition 3 *Let F be an implication base of a closure C on a finite set U, and let F consist of unary implications only. Then a partition $(A, \overline{A})$ of U is a strong decomposition pair of C_F iff A is a union of some connected components of G_F.*

Proof. First, notice that if A is a union of some connected components of G_F, then so is $\overline{A}$.

Let A be a union of some connected components of G_F. Then obviously A is closed and so is $\overline{A}$, i.e. $(A, \overline{A})$ is a strong decomposition pair by corollary 5.

Conversely, let $(A, \overline{A})$ be a strong decomposition pair of C_F. To finish the proof, we must show that if X is a connected component of G_F and $X \cap A \neq \emptyset$, then $X \subseteq A$. Let $x \in A \cap X$, and suppose there is $y \in X \cap \overline{A}$. Let $x_0 = x, x_n = y$ and $(x_0, x_1) \in V, (x_1, x_2) \in V, \ldots, (x_{n-1}, x_n)$ be a path in X from x to y. Then there exists at least one $i \in [1, n]$ such that $(x_i, x_{i+1}) \in V$ and $x_i \in A, x_{i+1} \in \overline{A}$. Since $(x_i, x_{i+1}) \in V$, either $x_i \to x_{i+1} \in F$ or $x_{i+1} \to x_i \in F$. In the first case by algorithm CLOSURE $x_{i+1} \in C_F(A) = A$, i.e. $A \cap \overline{A} \neq \emptyset$. In the second case $x_i \in C_F(\overline{A})$ and $A \cap \overline{A} \neq \emptyset$ again. This contradiction shows $X \subseteq A$. Thus A is a union of some connected components of G_F. Proposition is proved. $\square$

Consider the following algorithm UNARY DECOMPOSITION.

Algorithm UNARY DECOMPOSITION
Input: an implication system F over U consisting of unary implications.
Output: connected components $(X_1, \ldots, X_n)$ of G_F and their number n.
Method:
Construct G_F;
$n := 0$;
$U^0 := U$;
WHILE $U^0 \neq \emptyset$

DO
$n := n + 1$;
$X_n := \{x\}$ for $x \in U^0$;
WHILE there is $y \in U^0$, $y \notin X_n$ such that $(z, y) \in V$ for some $z \in X_n$
DO $X_n := X_n \cup \{y\}$ END;
$U^0 := U^0 - X_n$;
END;
RETURN$((X_1, \ldots, X_n, n))$.

Notice that this algorithm is polynomial since constructing transitive closure requires polynomial time.

Corollary 6 *Let F be an implication base of a closure C on a finite set U consisting of unary implications only. Then the strong decomposition pairs of C are exactly pairs $(\bigcup_{i\in I} X_i, \bigcup_{j\notin I} X_j)$, $I \subseteq \{1, ..., n\}$, where $(X_1, ..., X_n, n)$ is output of algorithm UNARY DECOMPOSITION when the input is F.* □

To construct a general algorithm for finding strong decomposition pairs we need some new concepts and two lemmas.

If we are given an implication system F, then $F' = \{X \to a : X \to Y \in F, a \in X - Y\}$ is an implication system satisfying $C_F = C_{F'}$. If the right hand sides of all the implications of an implication system are one-element sets, we will call this implication system *open* [Go]. The above remark shows that considering only open implication systems does not cause loss of generality. An implication system F will be called *nonredundant* if for every $f \in F : C_F \neq C_{F-f}$ [Ma,Wi]. Let F be an arbitrary implication system. Define $F^+ = \{X \to Y : Y \subseteq C_F(X)\}$. Then F^+ is an implication base of C_F too (it follows immediately from algorithm CLOSURE).

Lemma 3 *Let F be an open nonredundant implication base of a closure C on U. Then a partition $(A, \overline{A})$ is a strong decomposition pair of C iff the following hold:*
(i) $\forall X \to a \in F : X \subseteq A \Leftrightarrow a \in A$;
(ii) $\forall X \to a \in F : X \subseteq \overline{A} \Leftrightarrow a \in \overline{A}$.

Proof . Let $(A, \overline{A})$ be a strong decomposition pair, prove that (i) and (ii) hold. Let $X \to a \in F$ and $a \in A$. Then $a \in C_F(X)$, and $a \in C_F(X \cap A)$ because $(A, \overline{A})$ is a strong decomposition pair. According to algorithm CLOSURE, $X \to a$ can not be used to obtain $a \in C_F(X \cap A)$ if $X \not\subseteq A$. Hence, $C_F = C_{F-\{X \to a\}}$, and F is redundant. Thus, $X \subseteq A$. Obviously, if $X \subseteq A$ and $X \to a \in F$, then $a \in C_F(X) \subseteq A$. Therefore, (i) holds. Analogously, (ii) holds.

Let, conversely, (i) and (ii) hold. Then A and $\overline{A}$ are closed. Suppose $x \in C_F(X)$, and $x \in A$. Let $X_1 \to x_1, \ldots, X_k \to x_k$, $x_k = x$ be those implication which were used in algorithm CLOSURE to obtain $x \in C_F(X)$, ordered as they appeared in the algorithm. That means, $X_1 \subseteq X$, $X_2 \subseteq X_1 \cup \{x_1\}, \ldots, X_k \subseteq X_{k-1} \cup \{x_{k-1}\} \subseteq X \cup \{x_1, \ldots, x_{k-1}\}$. If for some $i : x_{i-1} \notin X_i$, then we can eliminate implication $X_{i-1} \to x_{i-1}$ from derivation $x \in C_F(X)$. Hence, we may suppose that no implication can be eliminated, and in this case $x_{i-1} \in X_i$ for $i \in [2, k]$. Since $x = x_k \in A$, by (i) $X_k \subseteq A$, and $x_{k-1} \in A$ because $x_{k-1} \in X_k$. Then by induction we obtain that $X_1 \cup \ldots \cup X_k \cup \{x_1, \ldots, x_k\} \subseteq A$, and according to algorithm CLOSURE $x \in C_F(X \cap A)$. Analogously, if $x \in \overline{A}$ then $x \in C_F(X \cap \overline{A})$. Thus, $(A, \overline{A})$ is a strong decomposition pair by corollary 4. Lemma is proved. □

Let F be an open implication system. Then F_T will stand for $\{x \to a : X \to a \in F, x \in X\}$.

Lemma 4 *Let F be a nonredundant open implication system. Then $(A, \overline{A})$ is a strong decomposition pair of C_F iff it is strong decomposition pair of C_{F_T}.*

Proof of lemma. Let $(A, \overline{A})$ be a strong decomposition pair of C_F. Consider $x \to a \in F_T$. Let $a \in A$. Since there is $X \to a \in F$, then $X \subseteq A$ and $x \in A$. Therefore, (i) and (ii) hold for F_T, and $(A, \overline{A})$ is a strong decomposition pair of C_{F_T}. Let, conversely, $(A, \overline{A})$ be a strong decomposition pair of C_{F_T}. Consider $X \to a \in F$. Let $a \in A$. Since for every $x \in X$: $x \to a \in F_T$ and $x \in A$, then $X \subseteq A$. Therefore, (i) and (ii) hold for F, and $(A, \overline{A})$ is a strong decomposition pair of C_F. □

Consider the following algorithm DECOMPOSITION.

Algorithm DECOMPOSITION
Input: an implication system F over U.
Output: a partition $(X_1, \ldots, X_n)$ of U
and the number n of its elements.
Uses algorithms: CLOSURE, UNARY DECOMPOSITION.
Method:
$F' := \{X \to a : X \to Y \in F, a \in Y - X\}$;
LOOP $X \to a \in F'$
IF $a \in$ CLOSURE $(F' - \{X \to a\}, X)$
THEN $F' := F' - \{X \to a\}$
END LOOP;
$F_T := \{x \to a : X \to a \in F', x \in X\}$;
$(X_1, \ldots, X_n, n) :=$ UNARY DECOMPOSITION(F_T);
RETURN$((X_1, \ldots, X_n, n))$.

The next result follows immediately from the previous lemmas, the fact that F' constructed in the LOOP in the above algorithm is an open nonredundant implication base of C_F (cf. [Ma]), and corollary 6.

Theorem 3 *Let F be an implication base of a closure C on a finite set U. Then the strong decomposition pairs of C are exactly the pairs $(\bigcup_{i \in I} X_i, \bigcup_{j \notin I} X_j)$, where $I \subseteq [1, n]$ and $(X_1, \ldots, X_n, n)$ is the output of algorithm DECOMPOSITION when the input is F.* □

Corollary 7 *Given an implication base F of a closure C on a finite set U, it takes polynomial time in the size of input to find a strong decomposition pair of C.* $\square$

In the rest of this section we present polynomial algorithm finding a representation of a distributive lattice as the direct product of directly indecomposable lattices.

Every finite distributive lattice $\mathcal{L}$ can be embedded in $\langle \mathbf{P}(U), \cap, \cup \rangle$ for some finite U (e.g. $U = J(\mathcal{L})$). Hence it is isomorphic to L_C where an implication base F of C consists of unary implications only. Therefore, each decomposition pair of C is strong, and for a strong decomposition pair $(A, \overline{A})$ the implication systems $F_A = \{x \to y \in F : x, y \in A\}$ and $F_{\overline{A}} = \{x \to y \in F : x, y \in \overline{A}\}$ are implication bases for $C\mid_A$ and $C\mid_{\overline{A}}$ respectively. Hence, applying algorithm UNARY DECOMPOSITION to F_A and $F_{\overline{A}}$ we obtain the direct product decompositions of $(A]$ and $(\overline{A}]$ and so on. Thus, applying UNARY DECOMPOSITION while it is possible we obtain a representation of $\mathcal{L}$ as the direct product of directly indecomposable lattices, if the input is F. Notice, that we also obtain a representation of closure C_F as the direct product of directly indecomposable closures.

The above algorithm is polynomial because it makes use of polynomial algorithm UNARY DECOMPOSITION no more than $\mid U \mid$ times.

However, a finite distributive lattice may not be represented by an implication base F consisting of unary implications. Now we consider three ways to represent a finite distributive lattice, and show how to construct an implication base consisting of unary implications in these cases.

First, if $\mathcal{L} \simeq L_C$ where C is given by its implication base F consisting of arbitrary implications, then for $F' = \{x \to y : X \to Y \in F, x \in X, y \in Y\}$ we have $C_F = C_{F'}$ (cf. [DLM2]).

It was proved in [Ri] that sublattices of $\langle \mathbf{P}(U), \cap, \cup \rangle$ containing $\{\emptyset\}$ and $\{U\}$ (we need these conditions because if L_C is a sublattice of $\langle \mathbf{P}(U), \cap, \cup \rangle$ then $\{U\} \in L_C$ and $\{\emptyset\} \in L_C$ by (C4)) and only they can be represented as

$$\mathcal{L} = \mathbf{P}(U) - \bigcup_{(x,y) \in P_{\mathcal{L}}} [x, U - y],$$

where $P_{\mathcal{L}} \subseteq U \times U$. Therefore, a sublattice of $\langle \mathbf{P}(U), \cap, \cup \rangle$ can be represented by a binary relation on U. Given $P_{\mathcal{L}} \subseteq U \times U$, let $F_{\mathcal{L}} = \{x \to y : (x, y) \in P_{\mathcal{L}}\}$. Then the lattice of closed sets of $C_{F_{\mathcal{L}}}$ is exactly $\mathcal{L}$, see [DLM1], [DLM2].

The most widely used way to represent a distributive lattice is that by a family of generating sets. If $X_1, \ldots, X_n \subseteq U$, let $L[X_1, \ldots, X_n]$ stand for the sublattice of $\langle \mathbf{P}(U), \cap, \cup \rangle$ generated by $X_1, \ldots, X_n$. Clearly, $L[X_1, \ldots, X_n]$ is distributive, and every finite distributive lattice is isomorphic to some $L[X_1, \ldots, X_n]$. The following proposition shows how to construct the family F.

Proposition 4 *Let $X_1, \ldots, X_n \subseteq U$. Suppose $x \to y \in F$ iff $\forall i \in [1, n] : x \in X_i \Rightarrow y \in X_i$. Then $L_{C_F} = L[X_1, \ldots, X_n]$.*

Proof. Let $X \in L[X_1, \ldots, X_n]$. Then $X = (X_1^1 \cap \ldots \cap X_{k_1}^1) \cup \ldots \cup (X_1^r \cap \ldots X_{k_r}^r)$ where $X_j^i \in \{X_1, \ldots, X_n\}$ for all $i \in [1, r], j \in [1, k_i]$. Suppose $x \to y \in F$ and $x \in X$. Then for some

$i \in [1,r]$ we have $x \in X_1^i \cap \ldots \cap X_{k_i}^i$ whence $y \in X_1^i \cap \ldots X_{k_i}^i$ and $y \in X$. Hence, $C_F(X) = X$, and $X \in L_{C_F}$.

Conversely, if $X \notin L[X_1, \ldots, X_n]$, then since $L[X_1, \ldots, X_n]$ is a sublattice of $\langle \mathbf{P}(U), \cap, \cup \rangle$ there are $a, b \in U$ such that $X \in [a, U-b]$ and $[a, U-b] \cap L[X_1, \ldots, X_n] = \emptyset$ by [Ri]. Then if $a \in X_i$ and $b \notin X_i$, we have $X_i \in [a, U-b]$ and $X_i \notin L[X_1, \ldots, X_n]$. Therefore, $a \to b \in F$, and $b \in C_F(X)$. Thus, $X \notin L_{C_F}$, and $L_{C_F} = L[X_1, \ldots, X_n]$. Proposition is proved. □

Summing up, we obtain

Corollary 8 *If a finite distributive lattice is represented by an implication base, or a binary relation, or a family of generating sets, there is an algorithm which is polynomial in the size of input and finds a representation of the lattice as the direct product of directly indecomposable lattices.* □

Notice that the results of this section dealing with the direct product decompositions of distributive lattices are related to those of [Fu].

We conclude this section by the remark showing that strong decomposition pairs can be obtained as optima of a simple problem of cluster analysis. Usually in clustering problem we have a function on pairs of elements which expresses either similarity or unsimilarity, and then, finding an optimum of some function we get clusters. Let p be a function that expresses similarity between elements of U, i.e. p is a real-valued function on $U \times U$, and we want to find a two-element partition $(A, \overline{A})$ of U. The typical criterion is
$F((A, \overline{A})) = \sum_{x \in A} \sum_{y \in \overline{A}} p(x,y) \longrightarrow min.$
(This criterion was used, for example, in [BH], but for the unsimilarities, i.e. maximum was to be found). Let F be an implication system over F. Let F be open and nonredundant. Suppose $p(x,y) = 1$ if there is $X \to y \in F$ such that $x \in X$, and $p(x,y) = 0$ otherwise. Then $F((A, \overline{A})) \geq 0$, and $F((A, \overline{A})) = 0$ iff $(A, \overline{A})$ is a strong decomposition pair by lemma 3. Therefore, strong decomposition pairs are exactly optimal solutions of the above clustering problem. More precisely, they are exactly global optima of F.

4 Atomistic lattices and closures

In this short and more "pure mathematical" section we are going to show that the characterization of the direct product decompositions of lattices of closed sets does work. That means, we can successfully apply this characterization to describe the direct decompositions of some lattices. In this section we will investigate some classes of *atomistic* lattices. A complete lattice is called atomistic if every element is a join of atoms[2]. Clearly, a complete atomistic lattice is the lattice of closed sets of a closure on the set of its atoms, and in turn this closure can be characterized as satisfying condition $C(x) = x$ for every element x.

[2]These lattices are called atomic in [Bi]. In [Gr] atomic lattices are those in which every element contains an atom. In this paper we prefer to make use of Grätzer's terminology.

Proposition 5 *Every decomposition pair of an atomistic closure is strong.*

Proof. Let C be an atomistic closure on U and $(A, \overline{A})$ its decomposition pair. Suppose there is $x \notin A \cup \overline{A}$. Then by (iii) of theorem 2 $x = C(x) = C(C(x) \cap (A \cup \overline{A})) = C(\emptyset) = \emptyset$ by (C4). This contradiction shows $A \cup \overline{A} = U$. □

One form of this proposition is well-known in matroid theory. Usually the product of matroids is introduced as the product of closures, and then it is proved that the products of matroids correspond exactly to the products of lattices of closed sets, see [Ai].

Now we apply theorem 2 to obtain a characterization of the direct product decompositions of lattices of sublattices and subsemilattices.

Let S be a semilattice, whose operation is denoted by $\cdot$. We think of S as being a join-semilattice, i.e. $x \leq y \Leftrightarrow x \cdot y = y$. Let $SubS$ stand for the lattices of all subsemilattices of S. Since $SubS$ is an algebraic lattice, it is the lattice of closed sets of an (algebraic) closure on the set of its atoms, i.e. S. In fact, given a subset $X \subseteq S$, its closure $C(X)$ is the least subsemilattice of S containing X. Let $(A, \overline{A})$ be a strong decomposition pair of this C. Suppose there are such $x \in A$ and $y \in \overline{A}$ that x and y are incomparable. Then $z = x \cdot y, x, y$ are distinct elements. If $X = \{x, y\}$, then $z \in C(X)$ and if we suppose without loss of generality $z \in A$ (because $A \cup \overline{A} = U$) then $z \in C(X) \cap A$ and $x = C(x) = C(X \cap A)$, i.e. (i) of theorem 2 fails. This contradiction shows that either $x \leq y$ or $y \leq x$. Since A and $\overline{A}$ are subsemilattices of S, and $(A] \simeq SubA$, $(\overline{A}] \simeq Sub\overline{A}$, we proved

Proposition 6 *Every direct product decomposition of lattice $SubS$ corresponds to an ordinal sum decomposition of S.* □

More precisely, if $SubS \simeq \prod_{i \in I} \mathcal{L}_i$, where all $\mathcal{L}_i$ are directly indecomposable, then S is isomorphic to the ordinal sum of semilattices S_i such that $SubS_i \simeq \mathcal{L}_i$ for all $i \in I$. In an arbitrary direct product decomposition $SubS \simeq \prod_{j \in J} \mathcal{M}_j$ each $\mathcal{M}_j$ is the lattice of subsemilattices of S^j, where S^j is the ordinal sum of some S_is.

This result was also announced in [DLM1], but the proof made use of distributive, standard and neutral element and some complex combinatorial structures. Here we obtained it almost immediately from theorem 2.

Notice, that if lattices are used instead of semilattices, all the above reasonings remain true if we forget about one operation. Thus, we get

Proposition 7 *Every direct product decomposition of a lattice $Sub\mathcal{L}$ of sublattices of $\mathcal{L}$ corresponds to an ordinal sum decomposition of $\mathcal{L}$.* □

This proposition was established in [Fi].

5 Direct product decomposition of relation schemes

Implication bases of closures are known under the name of *relation schemes* in the theory of relational databases. In this section we transfer the results of sections 2 and 3 to the relation schemes, with particular attention being paid to database problems such as a decomposition of a relation scheme into two or more relation schemes within one database scheme, normalization, finding mimimal keys and so on. We first introduce some terminology which is standard and can be found e.g. in [Ma]. Then we study the problem of decomposition and show that the most widely used normal forms are preserved under decomposition. We will also find the relationship between keys of a relation scheme and its subschemes determined by a decomposition. Finally, we investigate relationship between the decompositions of relation schemes and relation instances, i.e. relational databases themselves.

A *relation scheme* is a pair $\langle U, F \rangle$, where U is a finite set and F is an implication system. Elements of U are called attributes. They usually correspond to the attributes of a relational database, i.e. they are, e.g., name, date of birth, age, address an so on. Elements of F are called *functional dependencies* (*fd*s for short). For example, there could be a fd *name* $\rightarrow$ *address*, or a fd *date of birth* $\rightarrow$ *age*.

With each $a \in U$ associate its domain $dom(a)$. A *relation* over U is a subset $R \subset \prod_{a \in U} dom(a)$. We can think of R as being a set of mappings:
$R = \{t_1, \ldots, t_m\}$, $t_i : U \longrightarrow \bigcup_{a \in U} dom(a) \, : \, t_i(a) \in dom(a), i \in [1, m]$.
We say that R obeys a fd $X \rightarrow Y$ (or that this fd holds in R) if for every $t_i, t_j \in R$ the equality $t_i(X) = t_j(X)$ implies $t_i(Y) = t_j(Y)$ (by $t(X)$ we mean $\{t(x) : x \in X\}$). A relation R is said to be a *relation instance* of a relation scheme $\langle U, F \rangle$ if all the fds from F hold in R.

Let F_R stand for the set of all fds that hold in R. Then F_R satisfies two following properties:
(F1) $X \rightarrow Y \in F_R$ for all $Y \subseteq X$ (pseudoreflexivity);
(F2) $X \cup Z \rightarrow V \in F_R$ if $X \rightarrow Y \in F_R$ and $Y \cup Z \rightarrow V \in F_R$ (pseudotransitivity).
If we are given a set F of fds, let F^+ stand for the set of all fds that can be derived from F by using pseudoreflexivity and pseudotransitivity. Then $F_R^+ = F_R$ and F^+ thus defined coincides with F^+ defined in section 3 [Ma,DLM1,Wi]. Moreover, for every relation scheme $\langle U, F \rangle$ there is a relation R over U such that $F^+ = F_R$. This relation R is called an *Armstrong relation* of F [BDFS,MR1].

A set F of fds is called a *cover* of G if $F^+ = G^+$. A cover F is called *nonredundant* if for every $f \in F$ we have $f \notin (F - f)^+$. This concept of nonredundancy coincides with that defined in section 3. A cover is *open* [Go] if the right hand sides of its fds consist of one-element sets only. Every family F of fds has an open nonredundant cover. In fact, the first step of algorithm DECOMPOSITION from section 3 computes it.

A set X is called a *key* if $X \rightarrow U \in F^+$. A key is called minimal if each $Y \subset X$ is not a key. An attribute $a \in U$ is called *prime* if it belongs to a minimal key, and *nonprime* otherwise.

A relation scheme $\langle U, F \rangle$ is in

- *second normal form, or 2NF*, if $X \to a \notin F^+$ for $a \notin X$, a a nonprime attribute, and X a proper subset of a minimal key;
- *third normal form, or 3NF*, if $X \to a \notin F^+$ for $a \notin X$, a a nonprime and X a nonkey;
- *Boyce-Codd normal form, or BCNF*, if $X \to a \notin F^+$ for $a \notin X$ and X a nonkey.

A *database scheme* is a family of relation schemes $\langle U_1, F_1\rangle, \ldots, \langle U_k, F_k\rangle$ such that $U_1, \ldots, U_k$ are pairwise disjoint. An *instance* of a database scheme is a set $\{R_1, \ldots, R_k\}$, each R_i being an instance of $\langle U_i, F_i\rangle$.

Given a relation scheme $\langle U, F\rangle$, there is the closure C_F, and we can can consider its direct product decompositions. A direct product decomposition of the closure C_F will be also called a direct product decomposition of the relation scheme. Each direct product decomposition of C_F corresponds to a strong decomposition pair which will be also called a *strong decomposition pair of the relation scheme.*

Suppose $(A, \overline{A})$ is a strong decomposition pair of a relation scheme $\langle U, F\rangle$. Let F be open and nonredundant. Then for each $X \to a \in F$ either $X \cup a \subseteq A$ or $X \cup a \subseteq \overline{A}$. This means that attributes of A and $\overline{A}$ are "independent", i.e. no attribute of A functionally depends on a set of attributes of $\overline{A}$ and no attribute of $\overline{A}$ functionally depends on a set of attributes of A. Thus, we may suppose that actually we have two "independent" relation schemes $\langle A, F_A\rangle$ and $\langle \overline{A}, F_{\overline{A}}\rangle$, where $F_A = \{X \to a \in F : X \cup a \subseteq A\}$ and $F_{\overline{A}} = \{X \to a \in F : X \cup a \subseteq \overline{A}\}$. Clearly, $F_A \cup F_{\overline{A}} = F$ by lemma 3, i.e. we do not loose information decomposing a relation scheme into two relation schemes within one database scheme.

We have shown that the decompositions of a relation scheme do not cause the loss of information. However, it is important to know if we may or may not loose a nice structure of a database scheme when we decompose some of its relation schemes.

It is often required that a database scheme be in a normal form (second, third, or Boyce-Codd). We will show that the decompositions preserve these normal forms.

In sequel $\langle U, F\rangle$ will be an *arbitrary* relation scheme, and F_A, $F_{\overline{A}}$ will be covers of $\{X \to Y \in F^+ : X \cup Y \subseteq A\}$ and $\{X \to Y \in F^+ : X \cup Y \subseteq \overline{A}\}$ respectively. If A is closed, then the lattice of closed sets of C_{F_A} is the ideal $(A]$ of L_{C_F}. If F is open and nonredundant, and $(A, \overline{A})$ is a strong decomposition pair then we may choose F_A and $F_{\overline{A}}$ as we did above. We will need

Lemma 5 *Let $(A, \overline{A})$ be a strong decomposition pair of a relation scheme $\langle U, F\rangle$. Let $\mathcal{K}$ be a family of minimal keys of $\langle U, F\rangle$, and $\mathcal{K}_A$, $\mathcal{K}_{\overline{A}}$ the families of minimal keys of $\langle A, F_A\rangle$ and $\langle \overline{A}, F_{\overline{A}}\rangle$. Then $\mathcal{K} = \{K_1 \cup K_2 : K_1 \in \mathcal{K}_A, K_2 \in \mathcal{K}_{\overline{A}}\}$.*

Proof. If $K_1 \in \mathcal{K}_A$ and $K_2 \in \mathcal{K}_{\overline{A}}$, then obviously $K = K_1 \cup K_2$ is a key. Let $K' \subset K$ be a key, and let there be $a \in K - K'$. Suppose $a \in A$. Since K_1 is a minimal key of $\langle A, F_A\rangle$, then $C_F(K_1 - a) = Y \neq A$. Hence, $C_F(K') \subseteq C_F(K - a) = C_F((K_1 - a) \cup K_2) = C_F(Y \cup \overline{A}) =$

$Y \vee \overline{A} \neq U$ since A is neutral. This contradiction shows that K is a minimal key. By the analogous reasonings we show that if $K \in \mathcal{K}$, then $K \cap A \in \mathcal{K}_A$ and $K \cap \overline{A} \in \mathcal{K}_{\overline{A}}$. Lemma is proved. □

Theorem 4 *Let $\langle U, F\rangle$ be a relation scheme, and $(A, \overline{A})$ a decomposition pair. Then*
1) If $\langle U, F\rangle$ is in 2NF, then so are $\langle A, F_A\rangle$ and $\langle \overline{A}, F_{\overline{A}}\rangle$;
2) If $\langle U, F\rangle$ is in 3NF, then so are $\langle A, F_A\rangle$ and $\langle \overline{A}, F_{\overline{A}}\rangle$;
3) If $\langle U, F\rangle$ is in BCNF, then so are $\langle A, F_A\rangle$ and $\langle \overline{A}, F_{\overline{A}}\rangle$.

Proof. Notice that if $(A, \overline{A})$ is a decomposition pair, then according to the proof of lemma 5 any union of elements of $\mathcal{K}_A$ and $\mathcal{K}_{\overline{A}}$ is a minimal key of $\langle U, F\rangle$, since we never used $A \cup \overline{A} = U$ in the proof of lemma 5, but vice versa is not true in general.

Lemma 6 *Let $\langle U, F\rangle$ be a relation scheme, U_p the set of prime attributes, $(A, \overline{A})$ a decomposition pair, and $U_p(A), U_p(\overline{A})$ the sets of prime attributes of $\langle A, F_A\rangle$ and $\langle \overline{A}, F_{\overline{A}}\rangle$ respectively. Then $U_p(A) = U_p \cap A$ and $U_p(\overline{A}) = U_p \cap \overline{A}$.*

Proof of lemma. Let X be a coatom of $(A]$, i.e. a maximal closed set in $(A] - \{A\}$. Then $X \vee \overline{A}$ is a coatom in L_{C_F} (it follows immediately from lemma 2), and $(X \vee \overline{A}) \wedge A = X$. If Y is a coatom of L_{C_F}, then $Y \cap A$ is a coatom of $(A]$. Since the intersection of all coatoms of L_{C_F} is the set U_{np} of nonprime attributes [DT], then $U_{np}(A) = U_{np} \cap A$, whence $U_p(A) = U_p \cap A$. Lemma is proved.

1) Let $\langle U, F\rangle$ be in 2NF. We say that a closed set X is *prime* if $X = C_F(Y)$ where Y is a subset of a minimal key. According to [DLM2] a relation scheme is in 2NF iff for every prime set $X \neq U : [X \cap U_p, X] \subseteq L_{C_F}$. By lemma 6, it suffices to prove that for every X prime in $\langle A, F_A\rangle$, $X \neq A$, and every nonprime $a \in A, a \notin X$ the set $X - a$ is closed, because $X, X - a, a \in U_{np}(A)$ generate the interval $[X \cap U_p(A), X]$.

Let $X = C_F(Y)$ where $Y \subset Y'$, and $Y' \in \mathcal{K}_A$. If $Z \in \mathcal{K}_{\overline{A}}$, then $Y' \cup Z \in \mathcal{K}$, and $X' = X \vee \overline{A}$ is prime in $\langle U, F\rangle$ because $X' = C_F(Y \cup Z)$. Since A is neutral, $X' \cap A = X$. In particular, $a \notin X'$, and since $\langle U, F\rangle$ is in 2NF $X' - a \in L_{C_F}$. Hence, $X - a = (X' - a) \cap A \in L_{C_F}$, and $\langle A, F_A\rangle$ is in 2NF. Analogously we prove that $\langle \overline{A}, F_{\overline{A}}\rangle$ is in 2NF.

2) Let $\langle U, F\rangle$ be in 3NF. According to [DLM2] a relation scheme is in 3NF iff for every closed $X \neq U : [X \cap U_p, X] \subseteq L_{C_F}$. Again by lemma 6 it suffices to prove that for every closed $X \subset A$ and a nonprime $a \in A, a \notin X$ the set $X - a$ is closed. Let $Y = X \vee \overline{A} = C_F(X \cup \overline{A})$. Since A is neutral, $Y \cap A = X$, and $a \notin Y$. Therefore, $Y - a \in L_{C_F}$ because $\langle U, F\rangle$ is in 3NF and $Y \neq U$. Further, $X - a = (Y - a) \cap A \in L_{C_F}$. Since the lattice of closed sets of $\langle A, F_A\rangle$ is the ideal $(A]$ of L_{C_F}, $X - a$ is closed, and $\langle A, F_A\rangle$ is in 3NF. Analogously, $\langle \overline{A}, F_{\overline{A}}\rangle$ is in 3NF.

3) Let $\langle U, F\rangle$ be in BCNF. According to [DLM2], a relation scheme is in BCNF iff for every closed $X \neq U$ it holds: $[\emptyset, X] \subseteq L_{C_F}$. If $X \subset A$ is a closed set, then so is $X \vee \overline{A}$, and $X \vee \overline{A} \neq U$

because A is neutral. Hence, $[\emptyset, X] \subseteq [\emptyset, X \vee \overline{A}] \subseteq L_{C_F}$, and $[\emptyset, X] \subseteq (A]$. Thus, $\langle A, F_A \rangle$ is in BCNF, and so is $\langle \overline{A}, F_{\overline{A}} \rangle$. Theorem is completely proved. □

The result about BCNF has the simplest form if only strong decomposition pairs are taken into account. In fact, in this case nontrivial direct product decompositions do not exist. We say that a strong decomposition pair $(A, \overline{A})$ is *nontrivial* if both sets are nonempty. A relation scheme $\langle U, F \rangle$ is *trivial* if it consists only of trivial fds $X \to Y, Y \subseteq X$. In other words, $\langle U, F \rangle$ is trivial iff F has an empty cover.

Proposition 8 *Let $\langle U, F \rangle$ be a relation scheme in BCNF, and let $(A, \overline{A})$ be its nontrivial strong decomposition pair. Then $\langle U, F \rangle$ is trivial.*

Proof. Let $K_1, \ldots, K_k$ be the minimal keys of nontrivial relation scheme $\langle U, F \rangle$ in BCNF and let $(A, \overline{A})$ be a nontrivial strong decomposition pair, i.e. $A, \overline{A} \neq \emptyset$ (and $A, \overline{A} \neq U$). Since $(A, \overline{A})$ is a strong decomposition pair of C_F, for every i we have $C_F(K_i \cap A) = C_F(K_i) \cap A = A$. Since A is closed and $\langle U, F \rangle$ is in BCNF, $K_i \cap A$ is closed too because $A \neq U$, and $A = K_i \cap A$, i.e. $A \subseteq K_i$. Analogously $\overline{A} \subseteq K_i$ for all i. Therefore, $U = A \cup \overline{A} \subseteq K_i$. Hence, $\langle U, F \rangle$ has unique key, namely, U, and F consists only of trivial fds. □

By *decompositions* of a database scheme we will mean the following operations. Given a database scheme $\mathcal{S} = \{\langle U_1, F_1 \rangle, \ldots, \langle U_k, F_k \rangle\}$, and a strong decomposition pair $(A, \overline{A})$ of, say, $\langle U_i, F_i \rangle$, a primitive decomposition of $\mathcal{S}$ is a database scheme $\{\langle U_1, F_1 \rangle, \ldots, \langle U_{i-1}, F_{i-1} \rangle, \langle A, F_{iA} \rangle, \langle \overline{A}, F_{i\overline{A}} \rangle, \ldots, \langle U_k, F_k \rangle\}$. A decomposition of $\mathcal{S}$ is the result of some operations of primitive decomposition. We obtain immediately from the previous theorem

Corollary 9 *The decompositions of database schemes preserve normalization.* □

In the rest of the section we discuss the relationship between the decompositions of relation schemes and Armstrong relations. Two questions that arise here are the following. Given a relation scheme $\langle U, F \rangle$, its strong decomposition pair $(A, \overline{A})$ and Armstrong relations R_A and $R_{\overline{A}}$ of $\langle A, F_A \rangle$ and $\langle \overline{A}, F_{\overline{A}} \rangle$, how can we construct an Armstrong relation R of $\langle U, F \rangle$? And, if we are given an Armstrong relation R of $\langle U, F \rangle$, how can we construct R_A and $R_{\overline{A}}$?

The first question has been answered completely in [DFK] where construction of R is given. Great attention was paid to the problem of complexity in [DFK]. It is important that an Armstrong relation be small [BDFS,MR2], but in general it may have exponential size in the number of attributes and fds. However, the size of Armstrong relation of R is linear in the sizes of R_A and $R_{\overline{A}}$. In fact, let $s(F)$ be the size (the number of tuples, i.e. mappings t_i) of a minimal Armstrong relation of $\langle U, F \rangle$, and $s(F_A), s(F_{\overline{A}})$ be the sizes of minimal Armstrong relations of $\langle A, F_A \rangle$ and $\langle \overline{A}, F_{\overline{A}} \rangle$. If $(A, \overline{A})$ is a strong decomposition pair of $\langle U, F \rangle$, then $s(F) = s(F_A) + s(F_{\overline{A}}) - 1$ [DFK].

In this paper we answer the question concerning Armstrong relations R_A and $R_{\overline{A}}$. Let $R = \{t_1, \ldots, t_m\}$ be a relation over U, and $X \subseteq U$. Then $\Pi(R, X)$ is the projection of R onto X, i.e. $\{t_1|_X, \ldots, t_m|_X\}$.

Theorem 5 *Let $\langle U, F\rangle$ be a relation scheme and $(A, \overline{A})$ its strong decomposition pair. If R is an Armstrong relation of $\langle U, F\rangle$, then $\Pi(R, A)$ is an Armstrong relation of $\langle A, F_A\rangle$ and $\Pi(R, \overline{A})$ is an Armstrong relation of $\langle \overline{A}, F_{\overline{A}}\rangle$.*

Proof. It suffices to prove that $\Pi(R, A)$ is an Armstrong relation of $\langle A, F_A\rangle$. Introduce some definitions. Given a relation $R = \{t_1, \ldots, t_m\}$ over U, let $E_{ij} = \{a \in U : t_i(a) = t_j(a)\}$ and $E_R = \{E_{ij} : i, j \in [1, m]\}$. Let $L_F = L_{C_F}$ and $M(F)$ be the set of *meet-irreducible* elements of L_F. Then R is an Armstrong relation of $\langle U, F\rangle$ iff $M(F) \subseteq E_R \subseteq L_F$ [DT], cf. also [BDFS]. E_R is usually called the *equality set*.

Let R be an Armstrong relation of $\langle U, F\rangle$. Let E_R^A be the equality set of $\Pi(R, A)$. To prove that $\Pi(R, A)$ is an Armstrong relation of $\langle A, F_A\rangle$ we have to show that $E_R^A \subseteq (A]$ and each meet-irreducible element of $(A]$ is in E_R^A.

Let $X \in E_R^A$. Then for some $i, j \in [1, m]$ we have $X = \{a \in A : t_i(a) = t_j(a)\} = \{a \in U : t_i(a) = t_j(a)\} = E_{ij} \cap A$, where $E_{ij} \in E_R$. Since $E_R \subseteq L_F$, $X \in L_F$ and $X \in (A]$.

Let X be a meet-irreducible element in $(A]$. Let $Y = X \vee \overline{A}$, i.e. $Y = X \cup \overline{A}$ because $(A, \overline{A})$ is strong. Suppose Y is not meet-irreducible in L_F, i.e. $Y = Y_1 \cap Y_2$, $Y \neq Y_1, Y_2$. Then $X = (Y_1 \cap A) \cap (Y_2 \cap A)$ because $X = Y \cap A$. Since X is meet-irreducible in $(A]$, either $Y_1 \cap A = X$ or $Y_2 \cap A = X$. Suppose without loss of generality $X = Y_1 \cap A$. Then $\{X, Y, Y_1, A, U\}$ is a sublattice of L_F generated by A, Y, Y_1, and this sublattice is not distributive, which contradicts the neutrality of A. Hence, $Y \in M(F)$, and for some $i, j \in [1, m] : Y = E_{ij}$ because $M(F) \subseteq E_R$. Hence, $X = Y \cap A = E_{ij} \cap A = \{a \in A : t_i(a) = t_j(a)\} \in E_R^A$.

Thus, $\Pi(R, A)$ is an Armstrong relation of $\langle A, F_A\rangle$. Analogously, $\Pi(R, \overline{A})$ is an Armstrong relation of $\langle \overline{A}, F_{\overline{A}}\rangle$. Theorem is proved. □

6 Conclusion

In the paper we have studied the relationship between the direct product decompositions of closures and their lattices of closed sets. Every direct product decomposition of a closure corresponds to the one of its lattice of closed sets, but the direct product decompositions of lattice of closed sets may fail to correspond to the direct product decompositions of the closure.

Every direct product decomposition of a lattice of closed sets can be described by a pair of disjoint subsets of underlying set U on which the closure is defined, and the direct product decompositions of a closure correspond exactly to those pairs which are partitions of U.

If a closure is defined on a finite set by its implication base, there is a polynomial algorithm which computes a decomposition of the closure. This algorithm is based on one the computing of the direct product decompositions of topological closures whose lattices of closed sets are exactly distributive lattices.

The main characterization of the direct product decompositions of lattices of closed sets can be applied to find decompositions of some algebraic lattices, for example, lattices of sublattices and subsemilattices.

In the finite case the direct product decompositions of closures correspond to the decompositions of relational database schemes. Decomposing a scheme, we do not lose information. Decompositions of schemes can be described by projections of relations, and they preserve normalization, what is of practical importance, because it is often required that a database scheme be in a normal form.

One relevant problem is still open: given a poset, what is a characterization of its direct product decompositions? This problem is important, for example, in *domain theory* [GS] where a characterization of direct product decompositions of domains would be useful. There are also problems of finding representations analogous to implication bases, and of constructing algorithms to compute direct product decompositions. We plan to dedicate further research to these problems.

ACKNOWLEDGEMENT: The author is grateful to Peter Buneman for the useful discussions.

REFERENCES

[Ai] M.Aigner, *"Combinatorial Theory"*, Springer Verlag, Berlin, 1979.

[Ar] W.W.Armstrong, Dependency structure of data base relationships, *Information Processing 74*, North-Holland, Amsterdam (1974), 580-583.

[BDFS] C.Beeri, M.Dowd, R.Fagin, R.Statman, On the structure of Armstrong relations for functional dependencies, *J. of the ACM* 31 (1984), 30-46.

[BH] E.Boros, P.L.Hammer, On clustering problems with connected optima in Euclidean spaces, *Discrete Math.* 75 (1989), 81-88.

[Bi] G.Birkhoff, *"Lattice Theory"*, 3rd ed., AMS, Providence, RI, 1967.

[DFK] J.Demetrovics, Z.Füredi, G.O.H.Katona, Minimum matrix representation of closure operations, *Discrete Applied Math.* 11 (1985), 115-128.

[DLM1] J.Demetrovics, L.Libkin, I.B.Muchnik, Functional dependencies and the semilattices of closed classes, *Proc. of the second Symp. on Mathematical Fundamentals of Database Theory, Springer Lecture Notes in Comp. Sci.* 364 (1989), 136-147.

[DLM2] J.Demetrovics, L.Libkin, I.B.Muchnik, Functional dependencies in relational databases: a lattice point of view. Submitted to *Discrete Applied Math.*.

[DT] J.Demetrovics, V.D.Thi, Keys, antikeys, and prime attributes, *Annales Univ. Sci., Sect. Comp., Budapest* 8 (1987), 37-54.

[Fi] N.D.Filippov, Projectivity of lattices, *Amer. Math. Soc. Transl.* 96 (1970), 37-58.

[Fu] S.Fujishige, A decomposition of distributive lattices, *Discrete Math.* 55 (1985), 35-55.

[Go] G.Gottlob, On the size of nonredundant fd-covers, *Information Processing Letters* 24 (1987), 355-360.

[Gr] G.Grätzer, *"General Lattice Theory"*, Springer Verlag, Berlin, 1978.

[GS] C.Gunter, D.Scott, Semantic domains, In *"Handbook on Theoretical Computer Science"*, J. van Leeuwen, ed., North Holland, 1990, 633–674.

[Ma] D.Maier, *"The Theory of Relational Databases"*, Comp.Sci.Press, Rockville, MD, 1983.

[MR1] H.Mannila, K.-J.Räihä, Design by example: an application of Armstrong relations, *J. of Computer and System Sciences* 33 (1986), 126-141.

[MR2] H.Mannila, K.-J.Räihä, Practical algorithms for finding prime attributes and testing normal forms, *Proc. of the eighth Symp. on Principles of Database Systems*, ACM Press (1989), 128-133.

[Ri] I.Rival, Maximal sublattices of finite distributive lattices, *Proc. Amer. Math. Soc.* 44 (1974), 263-268.

[Wi] M.Wild, Implication bases for finite closure systems, Preprint No. 1210, Technische Hochschule Darmstadt, 1989.