www.ijecs.in
International Journal Of Engineering And Computer Science
Volume 14 Issue 11 November 2025, Page No. 27910-27917
ISSN: 2319-7242 DOI: 10.18535/ijecs/v14i11.5303

Derived conjunction and algorithm to close value block in the database model of block form

Trinh Ngoc Truc[1] , Dang Hong Hanh[1], Bui Tien Truong[1]
[1] Institute of Information Technology – Vietnam Women's Academy

Abstract:
In the database model of block form, from the results of the research on derived formulas, this article shows how to construct the derived conjunction to receive T_r as the truth block, the main results of this research has demonstrated the necessary and sufficient conditions for a logical formula to be expressed in terms of the derived equations. From these results, this article continues to propose the concepts of T_r value block, closed value block with multiplication & T^*, and propose algorithm to make closed value block in database model of block form, the results of this algorithm have found that the smallest truth block T^* contains the smallest truth block satisfying: T^* contains the element e unit and the z zero element and closes with the & operator. In special cases when the index set consists of 1 element, the volume degenerates into a relationship, this result coincides with the results researched by the authors in the relational database model.

Key words: Derived conjunction and algorithm to close value block in the database model of block form, block.

1. Introduction

The derivied formula in the form of $X \rightarrow Y$, where X and Y are logical conjunctions of finite variables, they keep a key role in the derivied motor of expert systems, in expressing data constraints of databases as well as algorithms that extract rules from the data warehourse. In the class of positive boolean dependencies, derivied formula are class of function dependencies proposed by Codd in 1970 [2], from which to apply mathematical concepts, logical formulas that have been formalized under Conjunction form of derivied formula. In the database model of block form, multivalue dependencies, general positive boolean dependencies have been researched, these results keep an important role in the design database of the block form. However, the research of derivied formula and conjunction have not been researched. Therefore, the objective of the article is to propose the concepts of derivied formula, derivied conjunctions, derivied construction algorithm from a given truth block, and algorithm to close value block.

2. The database model of block form

2.1 The block, slide of the block

Definition 2.1 ([1])

Let $R = (id; A_1, A_2,..., A_n)$ be a finite set of elements, where id is non-empty finite index set, $A_i(i=1..n)$ is the attribute. Each attribute $A_i(i=1..n)$ there is a corresponding value domain $dom(A_i)$. A block r on R, denoting $r(R)$ consists of a finite number of elements that each element is a family of mapping from the index set id to the value domain of the attribute $A_i(i=1..n)$.

$$t \in r(R) \Leftrightarrow t = \{ t^i : id \rightarrow dom(A_i)\}_{i=1..n} .$$

The block denotes $r(R)$ or $r(id; A_1, A_2,..., A_n)$, sometimes without fear of confusion it simply denotes r.

Definition 2.2([1])

Let $R = (id; A_1, A_2,..., A_n)$, $r(R)$ is a block on R. For each $x \in id$, $r(R_x)$ denotes is a block with $R_x = (\{x\}; A_1, A_2,..., A_n)$, such is:

$$t_x \in r(R_x) \Leftrightarrow t_x = \{t^i_x = t^i\}_{i=1..n} , \quad t \in r(R), t = \{ t^i : id \rightarrow dom(A_i)\}_{i=1..n} ,$$

x

Then $r(R_x)$ is called a slice of block $r(R)$ at point x.

2.2 Functional dependencies

Here for simplicity, The follwing notation is used:

$x^{(i)} = (x; A_i)$; $id^{(i)} = \{x^{(i)} \mid x \in id\}$.

$x^{(i)}(x \in id, i = 1..n)$ is called an index attribute of the block $R = (id; A_1,A_2,...,A_n)$.

Definition 2.3 ([1])

Let $R = (id; A_1,A_2,...,A_n)$, $r(R)$ is a block over R, $X, Y \in \bigcup_{i=1}^{n} id^{(i)}$, $X \to Y$ is a notation of function dependency. A block r satisfies $X \to Y$ if: $\forall t_1, t_2 \in r$ such is $t_1(X) = t_2(X)$ then $t_1(Y) = t_2(Y)$.

Definition 2.4 ([3])

Let block scheme $\alpha = (R,F)$, $R = (id; A_1, A_2,..., A_n)$, F is the set of functional dependencies on R. Then, the closure of F denoting F^+ is defined as follows:

$$F^+ = \{ X \to Y \mid F \Rightarrow X \to Y \}.$$

If $X = \{x^{(m)}\} \subseteq id^{(m)}$, $Y = \{y^{(k)}\} \subseteq id^{(k)}$ then functional dependency $X \to Y$ denotes simply $x^{(m)} \to y^{(k)}$.

The block r satisfies $x^{(m)} \to y^{(k)}$ if for any $t_1, t_2 \in r$ such is $t_1(x^{(m)}) = t_2(x^{(m)})$ then $t_1(y^{(k)}) = t_2(y^{(k)})$.

Where: $t_1(x^{(m)}) = t_1(x; A_m)$, $t_2(x^{(m)}) = t_2(x; A_m)$,

$t_1(y^{(k)}) = t_1(y; A_k)$, $t_2(y^{(k)}) = t_2(y; A_k)$.

Let Let $R = (id; A_1,A_2,...,A_n)$, subsets of functional dependency on R denote:

$$F_h = \left\{ X \to Y \mid X = \bigcup_{i \in A} x^{(i)}, Y = \bigcup_{j=B} x^{(j)}, A, B \subseteq \{1, 2, .., n\} \text{ and } x \in id \right\}$$

$$F_{hx} = F_h \Big|_{\bigcup_{i=1}^{n} x^{(i)}} = \left\{ X \to Y \in F_h \mid X, Y \subseteq \bigcup_{i=1}^{n} x^{(i)} \right\}$$

Definition 2.5 ([3])

Let block scheme $\alpha=(R,F_h)$, $R=(id; A_1, A_2,..., A_n)$, then F_h is called complete set of functionals dependenceis if F_{hx} is the same for all $x \in id$.

A more specific way:

$F_{hx}$is called the same for all $x \in id$, i.e: $\forall x, y \in id : M \to N \in F_{hx} \Leftrightarrow M' \to N' \in F_{hy}$ with M', N', respectively formed from M,N by replacing x by y.

2.3 Closure of the index sets attributes:

Definition 2.6 ([1])

Let block scheme $\alpha=(R,F)$, $R=(id; A_1, A_2,..., A_n)$, F is a set of functional dependencies on R.

For each $X \subseteq \bigcup_{i=1}^{n} id^{(i)}$, Closure of X for F denoting X^+ as follows:

$$X^+ = \left\{ x^{(i)}, x \in id, i = 1..n \mid X \to x^{(i)} \in F^+ \right\}.$$

The set of all subsets of $\bigcup_{i=1}^{n} id^{(i)}$ denotes a $Subset\left(\bigcup_{i=1}^{n} id^{(i)}\right)$.

3. Boolean formulas

3.1. Boolean formula

Definition 3.1 ([2])

Let $U = \{x_1, x_2, \ldots, x_n\}$ be a finite set of Boolean variables, B is Boolean value set, $B = \{0, 1\}$. Then the Boolean formulas, also known as logic formulas are constructed as follows:

(i) Each value 0/1 in B is a Boolean formula.

(ii) Each variable taking the value of U is a Boolean formula.

(iii) If a is a Boolean formula, then (a) is a Boolean formula.

(iv) If a and b are the Boolean formulas, then $a \wedge b$, $\neg a$ and $a \to b$ is a Boolean formula.

(v) Only the formula created by the rules from (i) - (iv) is Boolean formula.

$L(U)$ denotes the set of Boolean formulas built on a set of variables U .

Definition 3.2 ([2])

Each vector of the elements 0/1, $v = \{v_1, v_2, \dots, v_n\}$ in the space $B^n = B \times B \times \dots \times B$ is called a value assignment. Thus, with each Boolean formula $f \in L(U)$, it yields $f(v) = f(v_1, v_2, \dots, v_n)$ as the value of the formula f for value assignment v.

In case of confusion, it is understood that the symbols X performances are for the following subjects:

- A set of the attributes in U.
- A set of the logical variables in U.
- A Boolean formula is a logical conjunction of the variables in X.

On the other hand, if $X = \{B_1, B_2, \dots, B_m\} \subseteq U$, it denotes:

$\wedge X = B_1 \wedge B_2 \wedge \dots \wedge B_m$ and called the opportunity form.

$\vee X = B_1 \vee B_2 \vee \dots \vee B_m$ and called the recruitment form. The formula $f: Z \to V$ is called as:

- Derived formula if Z and V have the opportunity form, i.e. $f: \wedge Z \to \wedge V$.
- Strong derived formula if Z have the recruitment form and V have the opportunity form:

$$f: \vee Z \to \wedge V.$$

- Weak derived formula if Z have the opportunity form and V have the recruitment form:

$$f: \wedge Z \to \vee V.$$

- Duality derived formula if Z and V have the recruitment form:

$$f: \vee Z \to \vee V.$$

For every finite set of Boolean formula $F = \{f_1, f_2, \dots, f_m\}$ in $L(U)$, F is seen as a formula $F = f_1 \wedge f_2 \wedge \dots \wedge f_m$. Then it results in:

$$F(v) = f_1(v) \wedge f_2(v) \wedge \dots \wedge f_m(v).$$

3.2. Value table and truth table

For each formula f on U, the value table of f, denoting V_f contains $n + 1$ column, with n the first column contains the values of the variables in U, and the last column contains the value of f for each value assignment of the respective line. Thus, the value table contains 2^n line, n is the number of elements of U.

Definition 3.3 ([2])

Truth table of f, denoting T_f is the set of value assignment v such that $f(v)$ takes the value 1:

$$T_f = \{v \in B^n | f(v) = 1\}$$

Also, truth table T_F of a finite set of formulas F on U, is the intersection of the truth table of each member formulas in F.

$$T_F = \bigcap_{f \in F} T_f$$

Yields: $v \in T_F$ if and only if $\forall f \in F : f(v) = 1$.

3.3. Logic derived

Definition 3.4 ([2])

Let f, g be two Boolean formulas, said formula g derives from formula f logic, and symbols $f \models g$ if $T_f \subseteq T_g$. It is to say that f is equivalent to g and notation $f \equiv g$ if $T_f = T_g$.

With F and G in $L(U)$, said G logic derives from f, denoting that $F \models G$ if $T_F \subseteq T_G$.

Furthermore, it is to say F and G are equivalent, denoting $F \equiv G$ if $T_F = T_G$.

3.4. Positive Boolean formula

Definition 3.5 ([2])

The formula $f \in L(U)$ is called positive Boolean formulas if $f(e) = 1$ with $e = (1, 1, \dots, 1)$, and the set of all positive Boolean formulas over U denoting $P(U)$.

4. Research results

4.1. Derived formula in a block scheme

Definition 4.1.

Let $R = (id; A_1, A_2, \dots, A_n)$*, the derived formula on the block scheme is a formula with the form* $f: X \to Y; X, Y \subseteq \bigcup_{i=1}^{n} id^{(i)}$*, where X, Y are the conjunctions of the index attributes contained in it.*

Comment:

- Let X be an conjunction of logical variables, for each value assignment $v \in B^{n \times m} : X(v) = 1$ IFF $Set(v) \supseteq X$.

- Assuming $f : X \to Y$ is a derived formula on $\bigcup_{i=1}^{n} id^{(i)}$, then with each value assignment $v \in B^{n\times m}$ then:

$$f(v) = 1 \text{ IFF } (Set(v) \supseteq X) \Rightarrow (Set(v) \supseteq Y).$$

Let F be a set of derived formulas, $F = \{ f_1, f_2, \ldots, f_n \}$, F is considered to be a logical conjunction of component derived formulas $F = f_1 \wedge f_2 \wedge \ldots \wedge f_n$ and F is called derived conjunction

Definition 4.2

Let $R = (id; A_1, A_2, \ldots A_n)$, V is a set of value assigments on $\bigcup_{i=1}^{n} id^{(i)}$. Assuming $u, v \in V$, the multiplication operation of u and v, denoted $u \,\&\, v$, which is determined as follows:

If $u = \left(u_x^{(1)}, u_x^{(2)}, \ldots u_x^{(n)}\right)_{x\in id}$, $v = \left(v_x^{(1)}, v_x^{(2)}, \ldots v_x^{(n)}\right)_{x\in id}$ then $u \,\&\, v = \left(u_x^{(1)} \wedge v_x^1, u_x^{(2)} \wedge v_x^2, \ldots u_x^{(n)} \wedge v_x^n\right)_{x\in id}$

The product of an empty set of elements in V is the unit value assignment $e = (1,1,\ldots,1)$.

4.2. Properties of the close set family and the truth block

Definition 4.3

Let $R = (id; A_1, A_2, \ldots A_n)$, V is a set of value assigments on $\bigcup_{i=1}^{n} id^{(i)}$. The set of V value assignments is closed with multiplication & if V contains the product of every pair of elements in it, that is: $\forall u, v \in V : u \,\&\, v \in V$.

4.3. Algorithm for constructing derived conjunctions

Given a binary block *T*, the size of each element is $n\times m$ ($m=|id|$), containing *e* unit value assignments, *z* zero value assignments and closed with *&*. Then, the following XDF algorithm constructs the derived conjunction *F* on $R = (id; A_1, A_2, \ldots, A_n)$ taking *T* as the truth block.

XDF algorithm

Input: Binary block T □ $B^{n\times m}$, containing *e*, *z* and closed with &.

Output: Derived conjunction *F* on *R*, satisfying $T_F = T$.

Method

F:= □;

For each x □□id do

F_x:= □;

For each u □$B^n \setminus T_x$ do

X:= $Set(u)$;

$Y = \bigcap_{\substack{v\in T \\ Set(v)\supseteq X}} Set(v) \setminus X$;

F_x:=$F_x \cup \{X \to Y\}$;

endfor;

F:= $F \cup F_x$;

Endfor;

Return *F*

End XDF

To prove the correctness and complexity of the XDF algorithm, the following theorem is positive:

Theorem 4.1

Let a binary block T □ $B^{n\times m}$, containing e, z and closed with &, XDF algorithm calculates the set of devire formulas F takes T as the truth block.

Prove:

F is known as the set of derived formulas obtained through the XDF algorithm. To prove the correctness of the algorithm, we prove □v□T, $F(v) = 1$ và □v□ $B^{n\times m} \setminus T$, $F(v) = 0$.

Indeed, since *T* is a binary block containing *e, z* and is closed with &, in order to prove $F(v) = 1$, we only need to prove □v_x□ T_x, $F_x(v_x) = 1$, x □ id, similar to the case $F(v) = 0$, we just need to prove:

□v_x□ $B^n \setminus T_x$, $F_x(v_x) = 0$, x □ id, F_x, v_x is *F* and *v* respectively limited on the point slice x □ id.

Assuming v_x □ T_x , f_x: $X_x \rightarrow Y_x$ □ F_x và $Set(v_x) \supseteq X_x$, then according to the XDF algorithm it yields $u_x \in B^n \backslash T_x$ so that $X_x = Set(u_x)$ and $Y_x = \bigcap_{\substack{v \in T \\ Set(v_x) \supseteq X_x}} Set(v_x) \setminus X_x$.

Because v_x □ T_x và $Set(v_x) \supseteq X_x$ then $Set(v_x) \supseteq Y_x \Rightarrow f_x(v_x) = 1$.

Assuming □v_x□ $B^n \backslash T_x$, we need to show that in F_x exists f_x such that: $f_x(v_x) = 0$.

Considering f_x: X_x □ Y_x □ F_x is built from v_x according to the XDF algorithm, it yields: $X_x = Set(v_x)$ and $Y_x = \bigcap_{\substack{v \in T \\ Set(v_x) \supseteq X_x}} Set(v_x) \setminus X_x$. From the expression defining Y_x we see that X_x and Y_x do not intersect, otherwise $X_x = Set(v_x)$ infer $f_x(v_x) = 0$.

h is called the row number of block T, k is the row number of block $B^{n \times m} \backslash T$. Then, the XDF algorithm constructs k derived formulas. To construct each formula,we must perform h operations of the two sets, h takes the intersection of the two sets and a subtraction of the two sets. Each set of operations performed on n elements of the slice at x□id require complexity n. Finally, to regroup into F we need m unionization. In summary, it yields the complexity of the XDF algorithm is O($hkmn$).

4.4. The truth block T_r of the block

Definition 4.4

Let $R = (id; A_1, A_2, ...A_n)$, r is a block on the block scheme R, $u, v \in r$. In the case where the comparison of values between sets of the index attribute $x^{(i)}$ is equality, $x \in id$, then the truth block T_r of r is calculated as follows:

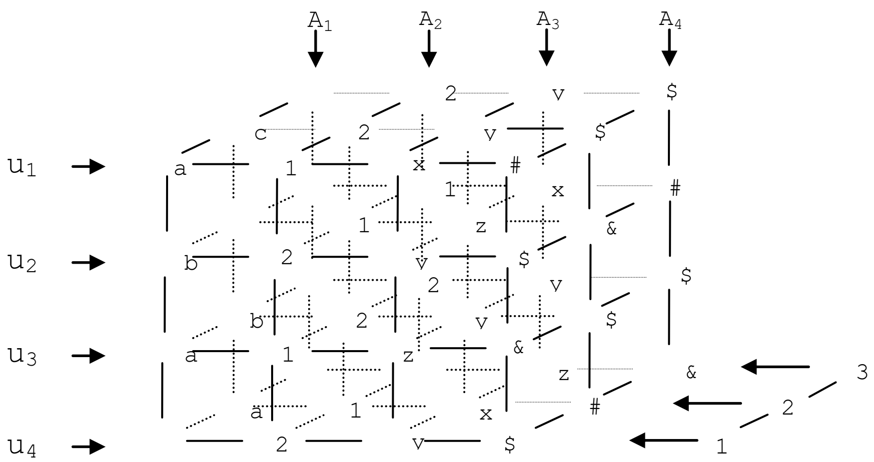
$T_r = \{\alpha(u,v) \mid u, v \in r\}$, in which $\alpha(u,v)$ is determined by the equation, namely if:

$u = (u.x^{(1)}, u.x^{(2)}, ..., u.x^{(n)}) \in r$, $v = (v.x^{(1)}, v.x^{(2)}, ..., v.x^{(n)}) \in r$

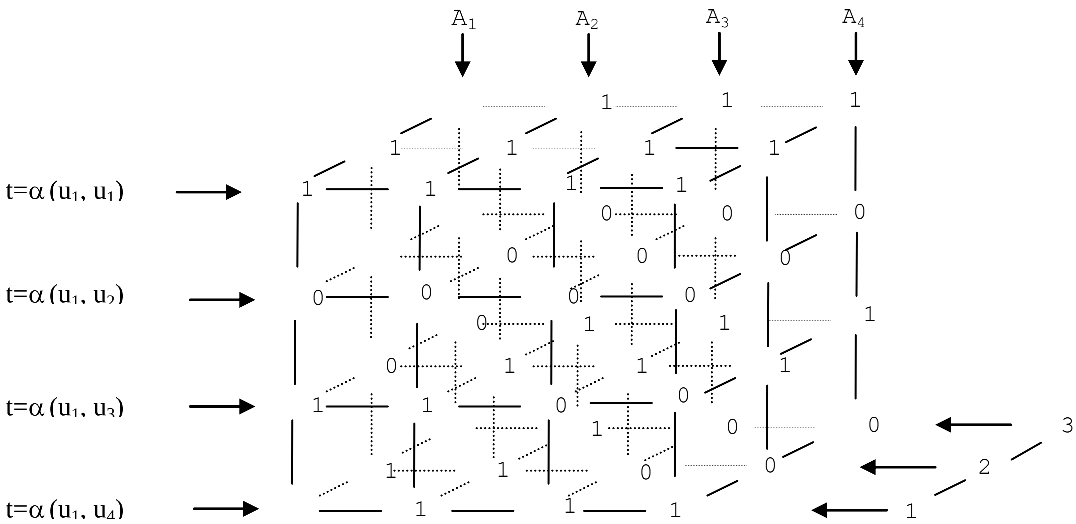
and assuming $\alpha(u,v) = t$; $t = \{t_1, t_2, ..., t_n\}\Big|_x$ then $t_i\big|_x = 1$ when $u.x^{(i)} = v.x^{(i)}$, $x \in id$, $i = 1..n$ and $t_i\big|_x = 0$ when $u.x^{(i)} \neq v.x^{(i)}$, $x \in id$, $i = 1..n$.

Example 1:

Let $R = (\{1,2,3\}; A_1, A_2, A_3, A_4)$, r a block of 4 elements over R with the values assignments as in the following block, calculate the T_r block:



Applying the formula to calculate T_r block according to Definition 4.4, we obtain the truth block T_r = ({1,2,3}; A_1, A_2, A_3, A_4) with the following results:



We find that, the product of the t (u_1, u_3) and t (u_1, u_4) elements in T_r has a non-T_r value on all $x \in$id slices.

With $x \in$id = 1 it yields: (1,1,0,0) & (0,1,1,1) = (0,1,0,0) $\notin T_r$

With $x \in$id = 2 it yields: (0,1,1,1) & (1,1,0,0) = (0,1,0,0) $\notin T_r$

With $x \in$id = 3 it yields: (0,1,1,1) & (1,1,0,0) = (0,1,0,0) $\notin T_r$

Therefore, T_r block does not close with &.

Definition 4.5

Let $R = (id; A_1, A_2, ...A_n)$, r is a block on the block scheme R, $u, v \in r$. In the case of id ={x}, the comparison of values between sets of the index attribute $x^{(i)}$ is still the equation, the truth block T_r of r becomes the truth table T_r of the relation r.

Example 2:

Let *r* as in example 1, with *id* = {1}, then block *r* degenerates into a relation *r* as follows:

r (A_1,	A_2,	A_3,	A_4)	And then	T_r (A_1,	A_2,	A_3,	A_4)
a	1	x	#		1	1	1	1
b	2	y	$		0	0	0	0
a	1	z	&		1	1	0	0
c	2	y	$		0	1	1	1

Comment: We see that, in general, block T_r does not close with multiplication &.

4.5. Algorithm closing value block 0/1

If block T_r on $B^{n \times m}$ does not close with &, T_r can be added with at least a number of elements to get block T^* so that the block satisfies the following two properties (i)-(ii):

(i) T^* contains unit element *e* and z zero element.

(ii) T^* closes with &.

The above process is called T_r block closure according to multiplication &.

Algorithm:

Algorithm Closed&

Format: Closed(T_r)

Input: *Block T_r on $B^{n \times m}$, $T_r = \{t_1, t_2, ..., t_d\}$*

Output: Smallest block T^* contains T_r and satisfies the properties

(i) T^* contains unit element *e* and *z* zero element.

(ii) T^* closes with &.

Method

$T^* := T_r$;

i:=0;

while *i* < *m* do

i:=*i*+1;

for each $x \in id$ do

for j:=1 to i-1 do

$t := t_i \& t_j$;

If t not_in T^* then

m:=m+1;

Add t to T^* as element t_m

Endif;

endfor;

endfor;

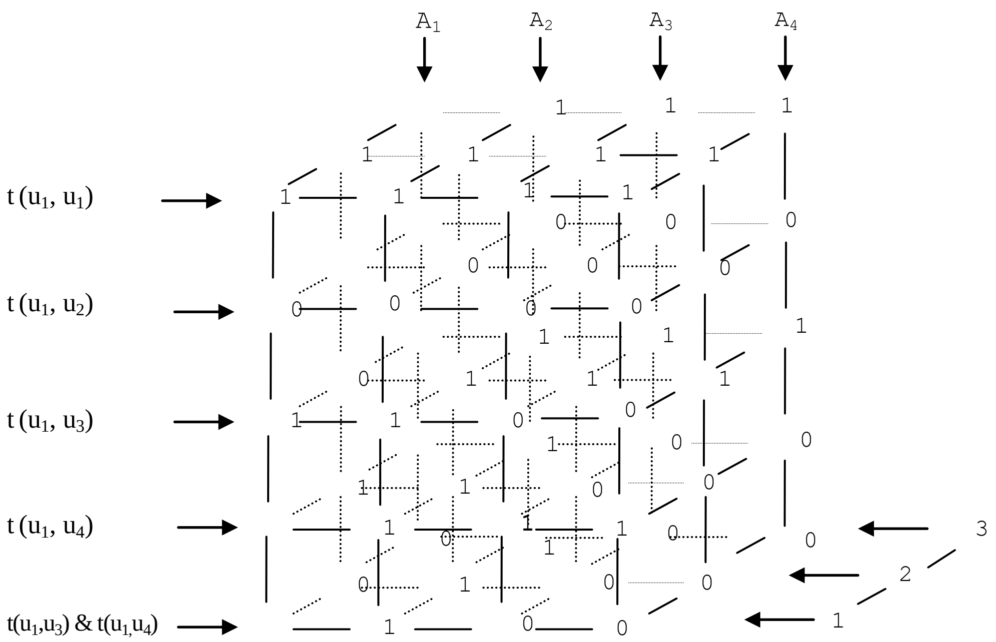
endwhile;

$T^* := T^* \cup \{e, z\}$;

Return T^*

End Closed&

Example 3: Let block r as in Example 1. Applying the algorithm closed block 0/1 as above, we get the t_5 element which is the product of t (u_1, u_3) & t (u_1, u_4) elements. Then Block T^* is obtained as follows:



From the above algorithm we have the following theorem:

Theorem 4.2

For each input of a block T_r in $B^{n\times m}$, the Closed & algorithm adding at least one element to get the block T^* which is a conjunctions lattice with multiplication &.

Prove:

Because multiplication & is had commutation properties, then table multiplication of every pair of elements crosses the main diagonal, so we only need to browse and fill in the lower half of the &. Specifically, consider the products of the $t_x^i \& t_x^j$; elements with each variable i from 1 to m and j varying from 1 to i-1, where m is the number of elements present in T^*.

Comment::

According to the Closed & algorithm, the elements added to T_r table to obtain the block T^* are the products of the elements in T_r. Therefore, they cannot be generators. We have the following consequences:

Consequence IV.2

If T^* = Closed &(T) then T contains the genes of the conjunction T^*.

5. Conclusion

Thus, with the results achieved, the paper has suggested the basic concepts of T_r value block, closed value block with multiplication & T^*, propose algorithm to construct the derived conjunction to receive T_r as the truth block, the algorithm to close the value block in the block data model. From the closed block algorithm, we have stated and proved that the *Closed &* algorithm added at least one element to get the block T^* is a chance with multiplication &. The results of this algorithm have found that the truth block T^* contains the smallest T_r block satisfying the property: T^* contains the unit element e and the z zero element and closes with the & operator. In special cases when the index set consists of 1 element, the volume degenerates into a relationship, this result coincides with the results researched by the authors in the relational data model.

6. References

1. Nguyen Xuan Huy, Trinh Dinh Thang , *Database model of block form*, Journal of Compurter Science and Cybernetics, P.14, S.3 (52-60), 1998.
2. Nguyen Xuan Huy, *Logical dependencies in database*, Statistical Publishing House, Hanoi, 2006.
3. Trinh Dinh Thang, Tran Minh Tuyen, *The translation of block scheme and the present problem of the closure, key in the database model of block forms,* 13th National Conference "Selected Problems about IT and Telecommunication", (276-286), Hung Yen, 19-20/08/2010.
4. Trinh Dinh Thang, Tran Minh Tuyen, *Key and key attributes set, non-key attributes set with translation of block schemes*, 14th National Conference "Selected Problems about IT and Telecommunication", (159-170), Can Tho 07-08/10/2011.
5. Trinh Dinh Thang, Tran Minh Tuyen, *Balancing scheme, minimum left hand side, and transformation block scheme,* 15th National Conference "Selected Problems about IT and Telecommunication", (174-179), Hanoi 03-04/12/2012.
6. Apostolos Benisis (2010), *Business Process Management: A Data Cube To Analyze Business Process Simulation Data For Decision Making Paperback***,** VDM Verlag Dr. Müller.
7. Brian Ciampa (2014), *The Data Warehouse Workshop: Providing Practical Experience to the Aspiring ETL Developer Paperback,* CreateSpace Independent Publishing Platform.
8. Ladjel Bellatreche, Mukesh K. Mohania (2014), *Data Warehousing and Knowledge Discovery:* 16th International Conference, DaWaK 2014, Munich, Germany, September 2-4, 2014. Proceedings (Lecture Notes ... Applications, incl. Internet/Web, and HCI) Paperback, Springer; 2014 edition.
9. Maurizio Rafanelli, M.Rafanelli (2003), *Multidimensional Databases: Problems and Solutions Hardcover*, Idea Group Publishing.
10. M.Rafanelli(Ed.) (2003), *Multidimensional Databases: Problems and Solutions,* Idea Group.
11. Qiang Yang, Joshua Zhexue Huang, Michaeng Ng (2003), *A data Cube Model for Prediction- Based Web Prefetching,* Journal of Intelligent Information Systems, Vol. 20, Issue 1, pp. 11-30.
12. R. Agrawal, A. Gupta, and S. Sarawagi (1997), *Modeling Multidimensional Databases,* In Proc. of 13 th, Int. Conf. on Data Engineering (ICDE) pages 232-243. IEEE press.