A Bit-Pattern-based Matrix Strategy for Efficient Iconic Indexing of Symbolic Pictures [1]

Ye-In Chang†, Bi-Yen Yang‡ and Wei-Horng Yeh†

†Dept. of Computer Science and Engineering
National Sun Yat-Sen University
Kaohsiung, Taiwan
Republic of China
{E-mail: changyi@cse.nsysu.edu.tw}
{Tel: 886-7-5252000 (ext. 4334)}
{Fax: 886-7-5254301}

‡Dept. of Applied Mathematics
National Sun Yat-Sen University
Kaohsiung, Taiwan
Republic of China

Abstract

In this paper, we propose an efficient iconic indexing strategy called *Bit-Pattern-based matrix* (BP matrix) for symbolic pictures, in which each spatial relationship between any two objects along the x-axis (or y-axis) is represented as a binary-bit pattern, and is recorded in a matrix. There are 12 bits in each bit pattern. When the bits in a certain subset of those 12 bits are set to 1, they denote a certain spatial relationship. Bit-wise-and/bit-wise-or operations are used for query processing; therefore, they are efficient enough as compared to the previous approaches. From our simulation, we show that the proposed BP matrix strategy requires shorter time for query processing than the Generalized Prime-Number-based (GPN) matrix strategy.

(*Keywords*: 2D string, 2D C-string, 9DLT, image databases, similarity retrieval, spatial reasoning)

[1] This research was supported in part by the National Science Council of Republic of China under Grant No. NSC-87-2213-E-110-014.

1 Introduction

In the Intelligent Image Database System (IIDS) proposed by S. K. Chang et al. [3, 4], they provided a high-level object-oriented search capability for *spatial reasoning*, where spatial reasoning means the inference of a consistent set of spatial relationships among the objects in an image. The IIDS is based on a pictorial data structure, called *2D string*, for indexing iconic spatial objects. However, the representation of 2D strings is not sufficient enough to describe pictures of arbitrary complexity completely [11], for example, pictures with (partly or completely) overlapping objects. For this reason, Lee and Hsu [11] proposed a *2D C-string* representation strategy. On the other hand, C. C. Chang et al. [5] proposed a *nine direction lower-triangular (9DLT) matrix*. In [8], Y. I. Chang et al. proposed a Generalized Prime-Number-based matrix strategy (denoted as the GPN matrix strategy), which combines the advantages of the 2D-C string and 9DLT matrix strategies and is an improved version of the Prime-Number-based matrix strategy [7]. In this GPN matrix strategy, each spatial relationship between any two objects is represented as a product of some prime numbers from a set of 12 prime numbers and is recorded in a matrix, and a module-based operation is used for the query processing. In recent years, a lot of new approaches focused on spatial relationship of similarity retrieval were published [1, 2, 9, 10, 13].

Although the GPN matrix strategy has improved the performance (in terms of query processing time) of the 2D C-string strategy, the GPN matrix strategy still requires a module-based operation to perform query processing. In this paper, we propose an efficient iconic indexing strategy called *Bit-Pattern-based matrix* (BP matrix) for symbolic pictures, in which each spatial relationship between any two objects along the x-axis (or y-axis) is represented as a binary-bit pattern, and is recorded in a matrix. There are 12 bits in each bit pattern. When the bits in a certain subset of those 12 bits are set to 1, they denote a certain spatial relationship. An efficient bit-wise-and/bit-wise-or operation is used to support spatial reasoning and similarity retrieval. From our simulation, we show that the proposed BP matrix strategy requires shorter time for query processing than the GPN matrix strategy.

The rest of the paper is organized as follows. Section 2 gives a brief description about

Table 1: Definitions of Lee's spatial operators (adapted from Ref. [12])

Notation	Condition	Meaning
A < B	end(A) < begin (B)	A disjoins B
A = B	begin(A) = begin(B) end(A) = end(B)	A is the same as B
A | B	end(A) = begin(B)	A is edge to edge with B
A % B	begin(A) < begin(B) end(A) > end(B)	A contains B and they have not the same bound
A [B	begin(A) = begin(B) end(A) > end(B)	A contains B and they have the same begin bound
A] B	begin(A) < begin(B) end(A) = end(B)	A contains B and they have the same end bound
A / B	begin(A) < begin(B) < end(A) < end(B)	A is partly overlapping with B

the 169 spatial relationships. Section 3 presents the proposed strategy. Section 4 shows the performance. Finally, Section 5 gives a conclusion.

2 Background

In Lee and Hsu's 2D C-sting [12], they presented the formal definition of the set of spatial operators as shown in Table 1, where the notation "begin(A)" denotes the value of begin-bound of object A and "end(A)" denotes the value of end-bound of object A [12]. According to the begin-bound and end-bound of the picture objects, spatial relationships between two enclosing rectangles along the x-axis (or y-axis) can be categorized into 13 types ignoring their length. Therefore, There are 169 types of spatial relationships between two rectangles in 2D space, as shown in Figure 1, where operator* denotes the inverse operator of the related operator, for example, $A < B$ implying $B <^* A$.

3 The BP Matrix Strategy

By observing the 169 spatial relationships in Figure 1, we can classify them into five spatial categories: *disjoin*, *join*, *contain*, *belong* and *part_overlap*, as shown in Figure 2, 3, 4, 5 and 6, respectively. Based on some observations from those tables, we design Bit-Pattern-based category rules.

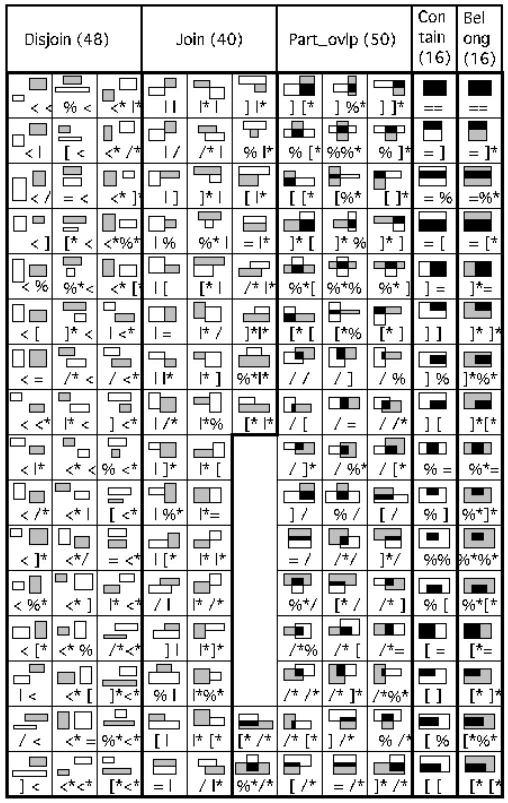


Figure 1: The 169 spatial relationship types of two objects (adapted from Ref. [12])

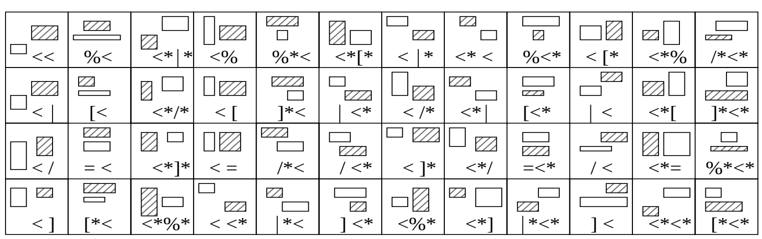


Figure 2: The 48 spatial relationships of category *disjoin*

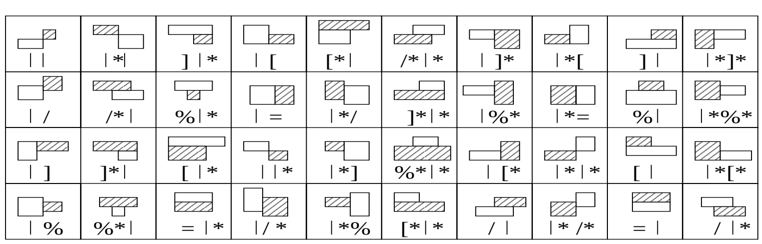


Figure 3: The 40 spatial relationships of category *join*

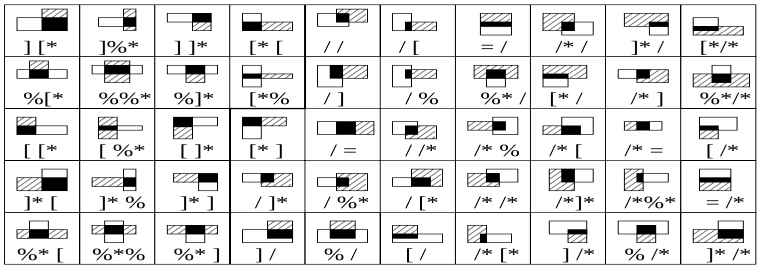


Figure 4: The 50 spatial relationships of category *part_ovlp*

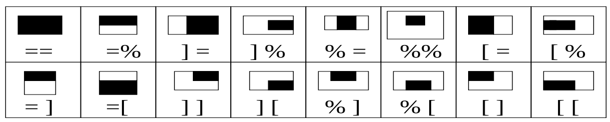


Figure 5: The 16 spatial relationships of category *contain*

3.1 Assignments of Spatial-Operator-Bit-Patterns (SOBP) for 13 Spatial Operators

Suppose A and B are two objects in a picture f, and the spatial relationship between them in terms of x-axis and y-axis is $(Ar^x_{A,B}B, Ar^y_{A,B}B)$, where $r^x_{A,B}$ and $r^y_{A,B}$ are the spatial operators in Table 1. The characteristics of those five spatial categories are described as follows.

1. Disjoin: One or both the $r^x_{A,B}$, $r^y_{A,B}$ spatial operators are in $\{<, <^*\}$.
2. Join: (a) None of the $r^x_{A,B}$, $r^y_{A,B}$ spatial operators is in $\{<, <^*\}$. And, (b) one or both the $r^x_{A,B}$, r^y_{AB} spatial operators are in $\{|, |^*\}$.
3. Contain: Both the $r^x_{A,B}$, $r^y_{A,B}$ spatial operators are in $\{=, \%,], [\}$.
4. Belong: Both the $r^x_{A,B}$, $r^x_{A,B}$ spatial operators are in $\{=, \%^*,]^*, [^*\}$.
5. Part_overlap: (a) One of the $r^x_{A,B}$, $r^y_{A,B}$ spatial operators is in $\{/, /^*\}$ and the other is in $\{\%, [,], /, =, \%^*, [^*,]^*, /^*\}$. Or (b) one of the $r^x_{A,B}$, $r^y_{A,B}$ spatial operators is in $\{\%,], [\}$ and the other is in $\{\%^*,]^*, [^*\}$.

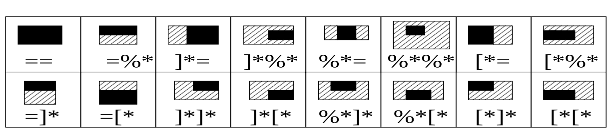


Figure 6: The 16 spatial relationships of category *belong*

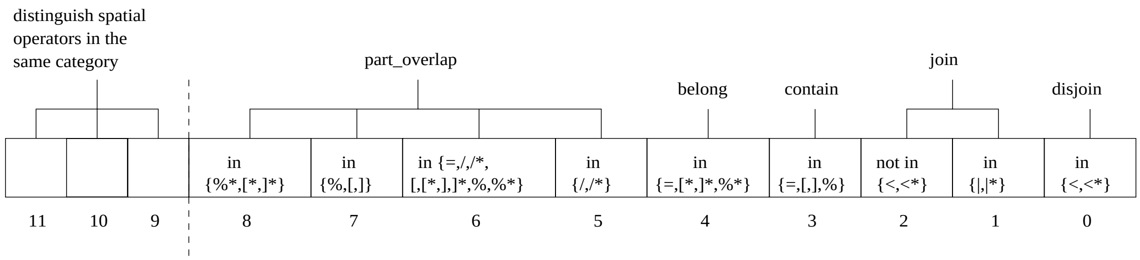


Figure 7: Meaning of those 12 bits

Based on the above observation, we can support efficient spatial reasoning by making use of the Bit-Pattern-based matrix representation. The major step is to assign each spatial operator a unique bit string $(b_{11}b_{10}b_9b_8b_7b_6b_5b_4b_3b_2b_1b_0)$ according to these five spatial categories. Figure 7 shows the meaning of those 12 bits. Suppose r is a spatial operator in the set $R = \{ <, <^*, |, |^*, [, [^*,],]^*, \%, \%^*, /, /^*, = \}$ and A, B are two objects in the symbolic picture. We define *sobp*(r) as the *spatial-operator-bit-pattern* of r with an initial value 000000000000. Here comes the steps of assignments.

(1) To classify the *disjoin* category, bit 0 is set. That is,

$$\text{sobp(r)} := \text{sobp(r)} \mid 000000000001, \qquad \forall r \in \{ <, <^* \}.$$

Therefore, the *Disjoin* category rule shown in Figure 8 can determine whether one or both the $r^x_{A,B}$, $r^y_{A,B}$ spatial operators are in the set $\{ <, <^* \}$.

(2) To classify the *join* category, bit 1 is set. That is,

$$\text{sobp(r)} := \text{sobp(r)} \mid 000000000010, \qquad \forall r \in \{ |, |^* \}.$$

This bit can be used to determine whether one or both the $r^x_{A,B}$, $r^y_{A,B}$ spatial operators are in the set $\{ |, |^* \}$. Moreover, none of the $r^x_{A,B}$, $r^y_{A,B}$ spatial operator should be in the set $\{ <, <^* \}$, so bit 2 is used. (In other words, both $r^x_{A,B}$, $r^y_{A,B}$ spatial operators should be in the set of $R \setminus \{ <, <^* \}$.) That is,

$$\text{sobp(r)} := \text{sobp(r)} \mid 000000000100, \qquad \forall r \in R \setminus \{ <, <^* \}.$$

Therefore, the *Join* category rule shown in Figure 8 can determine the *join* category.

(3) To classify the *contain* category, bit 3 is set. That is,

$$\text{sobp(r)} := \text{sobp(r)} \mid 000000001000, \qquad \forall r \in \{ =, \%,], [\}.$$

Therefore, the *Contain* category rule shown in Figure 8 can determine whether both the $r^x_{A,B}$, $r^y_{A,B}$ spatial operators are in the set $\{ =, \%,], [\}$.

(4) To classify the *belong* category, bit 4 is set. That is,

Disjoin	if ((sobp($r^x_{A,B}$) | sobp($r^y_{A,B}$)) & 000000000001) == 000000000001.
Join	if (((sobp($r^x_{A,B}$) & sobp($r^y_{A,B}$)) & 000000000100) | ((sobp($r^x_{A,B}$) | sobp($r^y_{A,B}$)) & 000000000010)) == 000000000110.
Contain	if ((sobp($r^x_{A,B}$) & sobp($r^y_{A,B}$)) & 000000001000) == 000000001000.
Belong	if ((sobp($r^x_{A,B}$) & sobp($r^y_{A,B}$)) & 000000010000) == 000000010000.
Part_overlap	if ((((sobp($r^x_{A,B}$) | sobp($r^y_{A,B}$)) & 000000100000) | ((sobp($r^x_{A,B}$) & sobp($r^y_{A,B}$)) & 000001000000)) == 000001100000) (* case (a) *) or (((sobp($r^x_{A,B}$) | sobp($r^y_{A,B}$)) & 000110000000) == 000110000000). (* case (b) *)

Figure 8: Bit-Pattern-based category rules

$$\text{sobp(r)} := \text{sobp(r)} \mid 000000010000, \qquad \forall r \in \{ =, \%^*,]^*, [^* \}.$$

Therefore, the *Belong* category rule shown in Figure 8 can determine whether both the $r^x_{A,B}$, $r^y_{A,B}$ spatial operators are in the set $\{ =, \%^*,]^*, [^* \}$.

(5) To classify the *part_overlap* category, let's consider the following two cases stated before. First, to determine whether one of the $r^x_{A,B}$, $r^y_{A,B}$ spatial operators is in the set $\{ /, /^* \}$, and the other is in the set $\{ \%, [,], /, =, \%^*, [^*,]^*, /^* \}$, bit 5 and bit 6 are used, respectively. That is,

$$\text{sobp(r)} := \text{sobp(r)} \mid 000000100000, \qquad \forall r \in \{ /, /^* \} = \text{K1}.$$

$$\text{sobp(r)} := \text{sobp(r)} \mid 000001000000, \qquad \forall r \in \{ \%, [,], /, =, \%^*, [^*,]^*, /^* \} = \text{K2}.$$

Therefore, the first test (before the "or" statement) of the *Part_overlap* category rule shown in Figure 8 can determine whether one of the $r^x_{A,B}$, $r^y_{A,B}$ spatial operators is in the set $\{ /, /^* \}$, and the other is in the set $\{ \%, [,], /, =, \%^*, \%^*, [^*,]^*, /^* \}$. Note that whenever bit 5 of r^x_{AB} (or r^y_{AB}) is set to 1, bit 6 of r^x_{AB} (or r^y_{AB}) is also set to 1 due to K1 $\subseteq$ K2. Therefore, we have to carefully eliminate the case that only one of r^x_{AB} and r^y_{AB} is in K1 while the other operator is not in K2. Table 2 shows all possible test results of bit 5 and bit 6 of r^x_{AB} and r^y_{AB}, where "*" denotes the special cases which we must be careful. (Note that the bit pattern "01" for b_6b_5 is impossible due to K1 $\subseteq$ K2.) For those special cases, for example, when b_6b_5 in r^x is 11 and b_6b_5 in r^y is 00, the final test result should be false. That is why the pattern for the test of bit 6 should be ((sobp($r^x_{A,B}$) & sobp($r^y_{A,B}$)) & 000001000000).

Table 2: The test results of bit 5 and bit 6

r^x (b_6 b_5)	r^y (b_6 b_5)	test1 := (r^x | r^y) & 01	test2 := (r^x & r^y) & 10	(test1 | test2) == 11
11	11	01	10	T
11	10	01	10	T
11	00	01	00	* F
10	11	01	10	T
10	10	00	10	F
10	00	00	00	F
00	11	01	00	* F
00	10	00	00	F
00	00	00	00	F

Similarly, we should be careful for the case in which b_6b_5 in r^x is 00 and b_6b_5 in r^y is 11.

Second, to determine whether one of the $r^x_{A,B}$, $r^y_{A,B}$ spatial operators is in the set $\{\, \%,], [\, \}$, and the other is in the set $\{\, \%^*,]^*, [^*\, \}$, bit 7 and bit 8 are used, respectively. That is,

$$\text{sobp(r)} := \text{sobp(r)} \mid 000010000000, \qquad \forall r \in \{\, \%,], [\, \}.$$

$$\text{sobp(r)} := \text{sobp(r)} \mid 000100000000, \qquad \forall r \in \{\, \%^*,]^*, [^*\, \}.$$

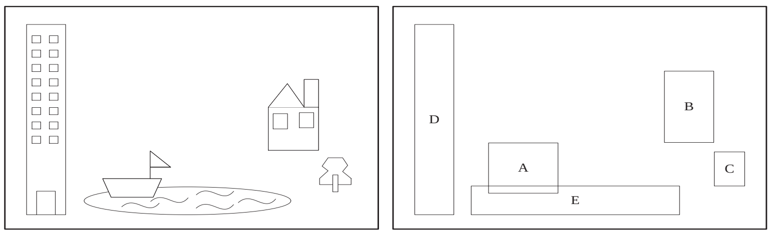
Therefore, the second test (after the "or" statement) of the *Part_overlap* category rule shown in Figure 8 can determine whether one of the $r^x_{A,B}$, $r^y_{A,B}$ spatial operators is in the set $\{\, \%,], [\, \}$, and the other is in the set $\{\, \%^*, [^*,]^*\, \}$.

According to the above descriptions, we have assigned each spatial operator a bit pattern which can be used to determine different spatial categories efficiently. However, in order to determine the spatial relationships between any two objects efficiently, we have to make each of the spatial-operator-bit-pattern unique. Therefore, we have to use more bits to distinguish those spatial operators which are in the same category. For the spatial operators in the *disjoin* category, we turn on bits 9 and 10 for spatial operators "<" and "<*", respectively. Similarly, for the spatial operators in the *join* category, we turn on bits 9 and 10 for spatial operators "|" and "|*", respectively. In the same way, for the spatial operators in the *part_overlap* category, we turn on bits 9 and 10 for spatial operators "/" and "/*", respectively. Moreover, for the spatial operators in the *contain* category, we turn on bits 9, 10 and 11 for spatial operators "%", "[" and "]", respectively. Note that when all those bit 9, bit 10 and bit 11 are 0, it implies the case of spatial operator

$<$: 001000000001
$|$: 001000000110
% : 001011001100
[: 010011001100
] : 100011001100
/ : 001001100100
= : 000001011100

$<^*$: 010000000001
$|^*$: 010000000110
%* : 001101010100
[* : 010101010100
]* : 100101010100
/* : 010001100100

Figure 9: The assignments of those 13 spatial operators



(a) (b)

Figure 10: An example: (a) an image; (b) its corresponding symbolic representation.

"=". Finally, for the spatial operators in the *belong* category, we turn on bits 9, 10 and 11 for spatial operators "%*", "[*" and "]*", respectively. Therefore, the assignments of spatial-operator-bit-pattern for these 13 spatial operators are shown in Figure 9.

3.2 Spatial Reasoning

For the symbolic picture shown in Figure 10, the corresponding *spatial matrix* S is shown as follows, where the spatial relationship between A and B along the x-axis (y-axis) is $A < B$ $(A|B)$:

$$S = \begin{array}{c} \\ A \\ B \\ C \\ D \\ E \end{array} \begin{array}{c} \begin{array}{ccccc} A & B & C & D & E \end{array} \\ \left[\begin{array}{ccccc} 0 & | & \% & \%^* & /^* \\ < & 0 & <^* & \%^* & <^* \\ < & | & 0 & \%^* & /^* \\ <^* & <^* & <^* & 0 & [\\ \%^* & /^* & <^* & < & 0 \end{array} \right] \end{array}$$

According to the assignments of spatial-operator-bit-pattern values for those 13 spatial operators described before, we can transform the spatial matrix S of f into a BP matrix T by replacing each spatial operator with its unique spatial-operator-bit-pattern as follows.

(Note that the size of the 2-dimension matrix is based on the number of objects shown in the corresponding picture.)

$$T = \begin{array}{c} \\ A \\ B \\ C \\ D \\ E \end{array} \begin{array}{c} \begin{array}{ccccc} A & B & C & D & E \end{array} \\ \left[\begin{array}{ccccc} 0 & 001000000110 & 001011001100 & 001101010100 & 010001100100 \\ 001000000001 & 0 & 010000000001 & 001101010100 & 010000000001 \\ 001000000001 & 001000000110 & 0 & 001101010100 & 010001100100 \\ 010000000001 & 010000000001 & 010000000001 & 0 & 010011001100 \\ 001101010100 & 010001100100 & 010000000001 & 001000000001 & 0 \end{array}\right] \end{array}$$

Based on the BP matrix, it is easy to retrieve the spatial relationships of each pair of objects along the x-axis and y-axis straightforwardly, since this information is recorded directly in the matrix. Moreover, the category of each pair of objects can be inferred by following the Bit-Pattern-based category rules as shown in Figure 8, in which only bit operations on the *sobp* value (the spatial-operator-bit-pattern) are needed.

3.3 Similarity Retrieval

The target of similarity retrieval is to retrieve the images that are similar to the query image.

Definition 1 *Picture f' is a type-i unit picture of f, if (1) all objects shown in f' must be shown in f, and (2) f' contains the two objects A and B, represented as x: $Ar^{x'}_{A,B}B$, y: $Ar^{y'}_{A,B}B$, A and B are also contained in f and the relationships between A and B in f are represented as x: $Ar^{x}_{A,B}B$, and y: $Ar^{y}_{A,B}B$, then,*

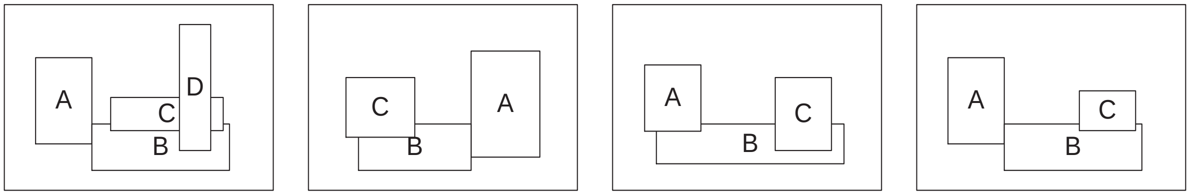
(type-0): $Category(r^x_{A,B}, r^y_{A,B}) = Category(r^{x'}_{A,B}, r^{y'}_{A,B})$;
(type-1): (type-0) and ($r^x_{A,B} = r^{x'}_{A,B}$ or $r^y_{A,B} = r^{y'}_{A,B}$);
(type-2): $r^x_{A,B} = r^{x'}_{A,B}$ and $r^y_{A,B} = r^{y'}_{A,B}$;

where $Category(r^x_{A,B}, r^y_{A,B})$ denotes the relationship category of the spatial relationship as shown in Table 1 [11, 12].

For example, in Figure 11, f_0 is a type-0 subpicture of f, f_1 is a type-1 subpicture of f, f_2 is a type-2 subpicture of f.

Definition 2 C is defined as follows.

C[i,j] = 1, if ((T[i,j] | T[j,i]) & 000000000001) == 000000000001;
C[i,j] = 2, if (((T[i,j] | T[j,i]) & 000000000010) | ((T[i, j] & T[j, i]) & 000000000100)) == 000000000110;



f f_0 f_1 f_2

Figure 11: similarity

C[i, j] = 3, if ((T[i,j] & T[j,i]) & 000000001000) == 000000001000;
C[i, j] = 4, if ((T[i,j] & T[j,i]) & 000000010000) == 000000010000;
C[i, j] = 5, if (((T[i, j] | T[j, i]) & 000000100000) | ((T[i,j] & T[j,i]) & 000001000000))
== 000001100000, or ((T[i,j] | T[j,i]) & 000110000000) == 000110000000,
$1 \leq i \leq m, 1 \leq j < i$.

That is, C[i, j] = 1, 2, 3, 4, 5 if the relationship between objects v_i and v_j is of the *disjoin*, *join*, *contain*, *belong* and *part_overlap* categories, respectively, by following the Bit-Pattern-based category rules.

The following three algorithms, *type-0*, *type-1*, *type-2* are used to determine whether two pictures are of type-0, type-1, type-2 similarity, respectively, given two BP matrix T_1 and T_2. (Note that before we do the similarity retrieval with the query picture T_2, we extract the sub-matrix T_1 from the related matrix of compared picture in the database with the same 2-dimension size as the query matrix T_2. Then, we can use the following algorithm to do the similarity retrieval.)

Algorithm (type-0)

(1) Following the Bit-Pattern-based category rules, find the category matrix C_1 and C_2 representing the two pictures f_1 and f_2, respectively.

(2) $C = C_1 - C_2$. If C is zero in the lower triangular matrix, these two pictures are of type-0 similarity; otherwise, there is no match.

Algorithm (type-1)

(1) Algorithm (type-0) passed.

(2) $T = T_1 - T_2$.

(3) $T^* = T(i,j) \times T(j,i) \qquad \forall \quad 1 \leq i \leq m, 1 \leq j < i.$
If T^* is zero in the lower triangular matrix (which implies that at least one of the spatial relationship along x-axis or y-axis is the same), these two pictures are of type-1 similarity; otherwise, there is no match.

Table 3: A comparison of the query processing time

	GPN	BP
type-0	0.105	0.070
type-1	0.106	0.070
type-2	0.010	0.005

Algorithm (type-2)

(1) $T = T_1 - T_2$. If T is zero, these two pictures are of type-2 similarity; otherwise, there is no match.

4 Simulation Study

In our simulation study, we consider the performance of query processing (of type-i similarity, $0 \leq i \leq 2$) for the GPN matrix and the proposed BP matrix strategies. To simplify our simulation, we let the maximum number of different objects appearing in the database be 20. For each object, it can appear in a picture with 100000 * 100000 points. We prepare 2000 pictures represented in the GPN matrix and the BP matrix representation in the database in advance, respectively. We consider case of 15 different objects appearing in each picture. For the query of type-i similarity, $0 \leq i \leq 2$, we compare one input query picture represented in the GPN matrix (or the BP matrix) with each of those prepared 2000 pictures in the database, respectively. Table 3 shows the simulation result (in terms of millisecond).

From this table, we observe that the proposed BP matrix strategy requires shorter time to process any of those kinds of queries than the GPN matrix strategy. This is because the GPN matrix strategy applies some module-based operations, it takes longer time than the bit-wise -and/bit-wise-or operations used in the BP matrix strategy.

For both of the strategies, to answer the type-2 query takes shorter time than to answer the type-0 query, since the algorithm to decide the category in the type-0 query is based on the spatial reasoning which is concerned in the type-2 query. Moreover, to answer the type-1 query may take a little longer time than to answer the type-0 query, since to satisfy the type-1 similarity must pass the test of type-0 similarity first.

There is another hash-table-based approach to similarity retrieval, for example, Chang

Table 4: A comparison of three strategies

	Proposed	C. C. Chang [6]	Sabharwal [14]
representation	matrix	hash table	hash table
spatial relationships	169	9	9
maintenance of database updates	incremental	reconstruction	reconstruction
similarity types	type-0, 1, 2	type-2	type-2

and Lee's strategy [6], and Sabharwal and Bhatia's strategy [14]. These two hash-table-based strategies perform well with complexity $O(n^2 \times \log l)$ when the database is never changed, where l is the number of pictures shown in database, and n is the number of objects shown in a query picture. However, when the data insertion/deletion occurs, these two hash-table-based strategies need to reconstruct the whole hash tables; while our proposed strategy only needs incremental update to the database. Note that in our proposed strategy, we just add the corresponding matrix to the image database, when there is a new image added to the database. That is, the maintenance of the update to the database is incremental based on our strategy. However, under the same situation, these two hash-table-based strategies must destroy the hash tables, recalculate the associated value for each object, and then reconstruct the hash tables, resulting in a large update cost. Moreover, our strategy can distinguish up to 169 spatial relationships, while the other two strategies can classify only 9 spatial relationships. Finally, our strategy can distinguish three levels of similarity retrieval, whereas the other two strategies can only support the exact match similarity retrieval which is the same as type-2 similarity. A summary of the comparison of our strategy with these two strategies is shown in Table 4.

5 Conclusion

In this paper, we have focused on the spatial relationship feature and have proposed an efficient iconic indexing strategy called *Bit-Pattern-based matrix* (BP matrix) for symbolic pictures, which combines the advantages of the *2D C-string* and the *9DLT matrix*. In the proposed strategy, we have designed each spatial operator a unique binary-bit pattern, and derived five Bit-Pattern-based category rules. Since those Bit-Pattern-based category rules are bit-wise operations, they are efficient enough as compared to the previous approaches. From our simulation, we have shown that the proposed BP matrix strategy has better performance than the GPN matrix strategy.

References

[1] A. F. Abate, M. Nappi, G. Tortora and M. Tucci, "IME: An Image Management Environment with Content-Based Access," *Image and Vision Computing,* Vol. 17, pp. 967–980, 1999.

[2] S. Berretti, A. Del Bimbo and E. Vicario, "Managing the Complexity of Matching in Retrieval by Spatial Arrangement," *Proc. Intern. Conf. Image Anal. Proc. (ICIAP'99), Venice, Italy,* Sept. 1999.

[3] S. K. Chang, Q. Y. Shi and C. W. Yan, "Iconic Indexing by 2-D Strings," *IEEE Trans. on Pattern Analysis and Machine Intelligence,* Vol. PAMI-9, No. 3, pp. 413–428, May 1987.

[4] S. K. Chang, C. W. Yan, D. C. Dimitroff and T. Arndt, "An Intelligent Image Database System," *IEEE Trans. on Software Eng.,* Vol. 14, No. 5, pp. 681–688, May 1988.

[5] C. C. Chang, "Spatial Match Retrieval of Symbolic Pictures," *Journal of Information Science and Engineering,* Vol. 7, No. 4, pp. 405–422, Dec. 1991.

[6] C. C. Chang and S. Y. Lee, "Retrival of Similar Pictures on Pictorial Databases," *Pattern Recognition,* Vol. 24, No. 7, pp. 675–680, July 1991.

[7] Y. I. Chang and B. Y. Yang, "A Prime-Number-Based Matrix Strategy for Efficient Iconic Indexing of Symbolic Pictures," *Pattern Recognition,* Vol. 30, No. 10, pp. 1–13, Oct. 1997.

[8] Y. I. Chang, B. Y. Yang and W. H. Yeh, "A Generalized Prime-Number-based Matrix Strategy for Efficient Iconic Indexing of Symbolic Pictures," *Pattern Recognition Letters,* Vol. 22, No. 5, pp. 657–666, May 2001.

[9] Y. I. Chang, H. Y. Ann and W. H. Yeh, "An Efficient Signature File Strategy for Similarity Retreival from Large Iconic Image Database," *Journal of Visual Languages and Computing,* Vol. 13, No. 2, pp. 117–147, April 2002.

[10] E. A. El-Kwae, M. Kabuka and A. Robust, "Framework for Content-Based Retrieval by Spatial Similarity in Image Database," *ACM Trans. on Information Systems,* Vol. 17, No. 2, pp. 174–198, 1999.

[11] S. Y. Lee and F. J. Hsu, "2D C-String: A New Spatial Knowledge Representation for Image Database Systems," *Pattern Recognition,* Vol. 23, No. 10, pp. 1077–1087, Oct. 1990.

[12] S. Y. Lee and F. J. Hsu, "Spatial Reasoning and Similarity Retrieval of Images Using 2D C-String Knowledge Representation," *Pattern Recognition,* Vol. 25, No. 3, pp. 305–318, March 1992.

[13] E. G. M. Petrakis, C. Faloutsos and D. Lin, "ImageMap: An Image Indexing Method Based on Spatial Similarity," *IEEE Trans. on Knowledge and Data Engineering,* 2001.

[14] C. L. Sabharwal and S. K. Bhatia, "Perfect Hash Table Algorithm for Image Databases Using Negative Associated Values," *Pattern Recognition,* Vol. 28, No. 7, pp. 1091–1101, July 1995.