非2値遺伝子のためのスキーマ分解*

高畠 一哉**・宮内 秀和**・岡田 三郎**

Schema Decomposition for Non-Binary Gene*

Kazuya TAKABATAKE**, Hidekazu MIYAUCHI** and Saburo OKADA**

In the theoretical analysis of genetic algorithms, WST (Walsh-Schema Transform) is a well-known method. WST is useful when we analyze genetic algorithms from the viewpoint of schema. However, WST can be used only when the gene is binary: the gene consists of only two types of symbols. We propose a WST-like method called OSD (Orthogonal Schema Decomposition) which can be used also for the non-binary gene. In this paper, the definition and the meaning of OSD, the fast algorithm for performing OSD, the relation between OSD and WST and some usages of OSD are shown.

1. Introduction

When the new offspring is made, the parents' schemata are combined to form the offspring's gene. Some schemata are inherited and others are not inherited. Genetic algorithms are the iteration of this process. For this reason, the analysis from the viewpoint of schema is an important issue when we study the mechanism of genetic algorithms.

WST (Walsh-Schema Transform; or also called partition coefficient transform)^{1)~3),5)} is a useful mathematical tool to analyze the genetic algorithms from the viewpoint of schema. However, WST can be used only when the gene is binary: the gene consists of only two types of symbols like 10110101. It is not necessary at all to use the binary genes in genetic algorithms. Actually the non-binary genes are used in the many applications of genetic algorithms⁴⁾. And also creature's genes are not binary.

In this paper a WST-like method called "Orthogonal Schema Decomposition (OSD)" which can be used for the non-binary gene is shown. In

2., we show the definition and the meaning of OSD. As WST has the fast transform algorithms¹⁾, OSD also has a fast decomposition algorithm. In 3., we show this fast algorithm. When the gene is binary, OSD and WST have the same meaning. Therefore, OSD is an expansion of WST. In 4., we review WST and show the relation between OSD and WST. In 5., we show some usages of OSD in the analysis of genetic algorithms: computing schema average^{1),3),5)}, evaluating the inheritance and evaluating the building block hypothesis³⁾. These usages are not new because we can use WST for the same analysis. But for the expansion to non-binary gene, we can do these analysis in many more cases than we could before.

2. Orthogonal schema decomposition

In the following sections, we will use the notations defined in Appendix.

2.1 Definition of OSD

We define G_t and H_t ($t \in T$) as

$$G_t = \{f \mid \forall x, y \in \Omega [x\$t = y\$t \Rightarrow f(x) = f(y)]\} \quad (1)$$

$$H_t = \begin{cases} G_{* \dots *}, & \text{if } t = * \dots * \\ G_t \cap \left(\sum_{u < t} G_u \right)^\perp, & \text{otherwise.} \end{cases} \quad (2)$$

$$\text{【Theorem 1】 } t \neq u \Rightarrow H_t \perp H_u, \quad (3)$$

* 原稿受付 1994年11月29日

** 通商産業省 工業技術院 中国工業技術研究所 Chugoku National Industrial Research Institute, AIST, MITI; 2-2-2 Hiro-Suehiro, Kure city, Hiroshima 737-01, JAPAN

Key Words: genetic algorithms, schema, Walsh-schema transform, function space, orthogonal decomposition.

(Proof of Theorem 1)

Let P_t be the projection to G_t . P_t is defined by

$$P_t f(x) = |x\rangle t|^{-1} \sum_{y \in x \S t} f(y). \quad (4)$$

Following lemma is clear.

【Lemma 1】 $f \in G_t \Rightarrow P_u f \in G_t \cap G_u$. (5)

Following lemma is clear by the definition of $\leq$ and the definition of G_t .

【Lemma 2】 $G_t \cap G_u = G_{t \wedge u}$. (6)

Let $f \in H_t$ and $g \in H_u$. We can assume $t \wedge u < t$ without loss of generality. Then

$$\begin{aligned} f \cdot g &= f \cdot P_t g \quad (\because f \in G_t) \\ &= 0 \quad (\because P_t g \in G_{t \wedge u}, f \perp G_{t \wedge u}). \end{aligned} \quad (7)$$

Q. E. D.

【Theorem 2】 $G_t = \sum_{u \leq t} H_u$. (8)

(Proof of Theorem 2)

$H_{*...*} = G_{*...*}$ is clear by definition. As the assumption of induction, assume that

$$\forall t < t' \left[G_t = \sum_{u \leq t} H_u \right]. \quad (9)$$

Then,

$$\begin{aligned} G_{t'} &= H_{t'} + \sum_{t < t'} G_t \\ &= H_{t'} + \sum_{t < t'} \sum_{u \leq t} H_u \\ &= \sum_{u \leq t'} H_u. \end{aligned} \quad (10)$$

Q. E. D.

【Corollary 1】 $V = \sum_{t \in T} H_t$. (11)

From these Theorems and corollary, any function $f \in V$ is uniquely decomposed to the functions as

$$f = \sum_{t \in T} f_t \text{ where } f_t \in H_t. \quad (12)$$

f_t 's are orthogonal each other. This decomposition is OSD.

2.2 Meaning of OSD

When $f \in G_t$, the value of f depends on only figures located at the locus where t has $\%$. Let $O(f)$ be the order of function f defined by

$$O(f) = \min \{O(t) \mid f \in G_t\}. \quad (13)$$

OSD is the successive approximation through low-order functions to high-order functions. For example, in the case of $\Omega = \Omega_0 \times \Omega_1$:

- (1) Choose $f_{**} \in G_{**}$ (=constant function) so that $\|f - f_{**}\|$ is minimized.
- (2) Choose $f_{*\%} \in G_{*\%}$, $f_{\%*} \in G_{\%*}$ so that $\|f - f_{**} - f_{*\%}\|$ and $\|f - f_{**} - f_{\%*}\|$ are minimized.
- (3) Let $f_{\% \%} = f - f_{**} - f_{*\%} - f_{\%*}$.

When v is a given vector and A is a subspace, for minimizing $\|v - a\|$ ($a \in A$), we must choose a as $a = Pv$ where P is the projection to A . Then the procedures (1), (2) and (3) are equal to :

- (1) Let $f_{**} = P_{**} f$.
- (2) Let $f_{*\%} = P_{*\%} (f - f_{**})$ and $f_{\%*} = P_{\%*} (f - f_{**})$.
- (3) Let $f_{\% \%} = f - f_{**} - f_{*\%} - f_{\%*}$.

We show an example of OSD in the case that $\Omega = \{0, 1, 2\}^2$ in Fig. 1.

Fig. 1 Example of OSD

3. Fast algorithm for OSD

We can perform OSD by the algorithm shown in the last section. However, this algorithm is for explaining the meaning of OSD, not for practical computation of OSD. OSD has a fast algorithm as WST has the fast algorithm¹⁾ like Cooley - Turkey's FFT. For practical computation of OSD, the algorithm FOSD shown in this section is recommended.

Fig. 2 shows the principle of FOSD in the case that $\Omega = \Omega_0 \times \Omega_1$.

Fig. 2 Principle of FOSD

[Corollary 2] $[\] t \leq u \Rightarrow H_t \perp G_u$. (14)

Therefore, $f_{**} + f_{*\%}$ and $f_{*\%} + f_{\%*}$ are gotten by decomposing f to $G_{*\%}$ component and $G_{*\%}^\perp$ component. And then, all functions f_t are gotten by decomposing these two functions to $G_{*\%}$ component and $G_{*\%}^\perp$ component.

To represent f_t in computers, we do not need to memorize values $f_t(x)$ for all $x \in \Omega$. Because as far as $x \in x' \$ t$, $f_t(x)$ has the same value. Let $F: S \rightarrow R$ be the function defined by

$$F(x \$ t) = f_t(x). \quad (15)$$

Then F has all information about all f_t 's. Using F , we can perform OSD as illustrated in Fig. 3.

Fig. 4 shows the sample program of FOSD where gene length is two, and Fig. 5 shows it where gene length is arbitrary N .

$x_0 \backslash x_1$	0	1	2	*
0	0	1	5	
1	4	3	8	
2	8	8	8	
*				

↓ Decomposition to $G_{*\%}$ and $G_{*\%}^\perp$

$f_{*\%} + f_{*\%}^\perp$				
-4	-3	-2		
0	-1	1		
4	4	1		
4	4	7		

↓ Decomposition to $G_{*\%}$ and $G_{*\%}^\perp$

$f_{*\%}$	$f_{*\%}^\perp$			
-1	0	1	-3	
0	-1	1	0	
1	1	-2	3	
-1	-1	2	5	

Fig. 3 Example of FOSD

$/ * \Omega_0 = \{0, \dots, OM0-1\}$ and $\Omega_1 = \{0, \dots, OM1-1\}$. Assume that OM0 and OM1 have been already defined as constants. $F[x_0][x_1] = F(x_0 x_1)$ where $x_i = * \Leftrightarrow x_i = |\Omega_i|$. Assume that $F[\][\]$ is initialized as

$$F[x_0][x_1] = \begin{cases} f(x_0 x_1) & x_0 < |\Omega_0| \text{ and } x_1 < |\Omega_1| \\ \text{arbitrary value} & \text{otherwise.} \end{cases}$$

```

*/
FOSD (F) double F [OM0+1] [OM1+1]; {
int x0, x1; double avr;

/* Decomposition to  $G_{*\%}$  and its orthocomplement */
for(x1 = 0; x1 < OM1; x1++) {
  for (x0 = 0, avr = 0; x0 < OM0; x0++) avr += F[x0][x1];
  F[OM0][x1] = avr / OM0;
  for (x0 = 0; x0 < OM0; x0++) F[x0][x1] -= avr;
}

/* Decomposition to  $G_{*\%}^\perp$  and its orthocomplement */
for(x0 = 0; x0 < OM0; x0++) {
  for(x1 = 0, avr = 0; x1 < OM1; x1++) avr += F[x0][x1];
  F[x0][OM1] = avr / OM1;
  for(x1 = 0; x1 < OM1; x1++) F[x0][x1] -= avr;
}
}

```

Fig. 4 Example of the program (gene length=2)

$/ * N (= N)$ is the gene length. $\Omega_i = \{0, \dots, OM[i]-1\}$. $F[\]$ denotes the decomposed function F . Schema $x_0 \dots x_{N-1}$ corresponds to the $F[\]$'s suffix $s = a_0 x_0 + \dots + a_{N-1} x_{N-1}$ where $a_0 = 1$, $a_{i+1} = a_i(|\Omega_i| + 1)$ and $x_i = * \Leftrightarrow x_i = |\Omega_i|$. Assume that $F[\]$ is initialized as

$$F[s] = \begin{cases} f(x_0 \dots x_{N-1}) & \forall i [x_i < |\Omega_i|] \\ \text{arbitrary value} & \text{otherwise.} \end{cases}$$

Following program treats schema $l \dots l x u \dots u$ where i is the locus of x ; lower determines $l \dots l$ and upper determines $u \dots u$

```

*/
FOSD (N, OM, F) int N, OM [ ]; double F [ ]; {
int lower, x, upper i, ai, aipl, S; double avr;
for(i=0, S=1; i < N; i++) S *= OM[i] + 1; /* S = |S| */
for(i=0, ai=1; i < N; i++) {
  aipl = ai * (OM[i] + 1); /* aipl =  $a_{i+1}$  */
  for(upper=0; upper < S; upper += aipl) { /* loop for all  $u \dots u$  */
    for(lower=0; lower < ai; lower++) { /* loop for all  $l \dots l$  */
      for(x=0, avr=0; x < OM[i]; x++) avr += F[upper+lower+ai*x];
      F[upper+lower+ai*x] = avr / OM[i];
      for(x=0; x < OM[i]; x++) F[upper+lower+ai*x] -= avr;
    }
  }
  ai = aipl;
}
}

```

Fig. 5 Example of the program (gene length= N)

4. Relation between OSD and WST

4.1 Review of WST

Let $\Omega = \{0, 1\}^N$ in this subsection. Let $\text{No1}(s)$ be the number of 1 contained in $s \in S$. Let ϕ_t be the function defined by

$$\phi_t(x) = (-1)^{\text{No1}(x\$t)}. \quad (16)$$

$\{\phi_t\}$ is called the basis of WST. Fig. 6 shows the $\{\phi_t\}$ in the case that gene length is three.

Fig. 6 Basis of WST

[Theorem 1'] $t \neq u \Rightarrow \phi_t \perp \phi_u$.

$\{\phi_t\}$ ($t \in T$) is linearly independent. $\phi_t \in G_t$. $\dim(G_t) = |\{u \mid u \leq t\}| = 2^{O(t)}$. Therefore,

[Theorem 2'] The subspace spanned by $\{\phi_u\}$ ($u \leq t$) is G_t .

These theorems correspond to **Theorems 1** and **2** about OSD. For any function $f \in V$, there exist unique $\{w_t \in \mathbf{R}\}$ ($t \in T$) such that

$$f = \sum_{t \in T} w_t \phi_t. \quad (17)$$

This is WST.

4.2 Relation between OSD and WST

[Theorem 3] $\phi_t \in H_t$. (18)

(Proof of Theorem 3)

$\sum_{u < t} G_u$ is the subspace spanned by $\{\phi_u\}$ ($u < t$)

by **Theorem 2'**. $\phi_t \in G_t$ and

$$\forall u < t [\phi_u \perp \phi_t]. \quad (19)$$

therefore $\phi_t \in H_t$ by the definition of H_t .

Q. E. D.

[Theorem 4] $\dim(H_t) = 1$. (20)

(Proof of Theorem 4)

By **Theorems 1** and **2**

$$\dim(V) = \sum_{t \in T} \dim(H_t) = 2^N = |T|. \quad (21)$$

By **Theorem 3**, $\dim(H_t) \geq 1$. Therefore $\dim(H_t) = 1$.

Q. E. D.

By these theorems,

$$\forall t \exists w_t \in \mathbf{R} [f_t = w_t \phi_t]. \quad (22)$$

These w_t 's are same that w_t 's in eq. (17).

5. How to use OSD

5.1 Computing schema average

The schema average^{(1), (3), (5)} is a very basic measure in the schema analysis. It is defined by

$$A(s) = |s|^{-1} \sum_{x \in s} f(x). \quad (23)$$

Therefore,

$$\begin{aligned} A(x\$t) &= |x\$t|^{-1} \sum_{y \in x\$t} f(y) \\ &= P_t f(x) \quad (\because \text{eq. (4)}) \\ &= \sum_{u \leq t} f_u(x) \quad (\because [\sqcap u \leq t] \Rightarrow H_u \perp G_t). \end{aligned} \quad (24)$$

Using eq. (15), we can rewrite eq. (24) in the short form :

$$A(s) = \sum_{u \leq s} F(u). \quad (25)$$

5.2 Evaluating the inheritance

Using OSD, we can decompose the value $f(x)$ into $F(s)$'s where schema $s \leq x$. And we know where s comes from. Assume that there are two genes called *dad* and *mom*, and their schema is combined to form the gene *kid*. If $s \leq \text{dad}$, s comes from *dad*. If $s \leq \text{mom}$, s comes from *mom*. For example, let *dad* = 2685, *mom* = 1646 and *kid* = 2646.

$$\begin{aligned} f(2646) &= F(2***)+F(26**) \\ &\quad +F(****)+F(*6**) \\ &\quad +F(**4*)+F(***6) \\ &\quad +F(*64*)+F(*6*6) \\ &\quad +F(**46)+F(*646) \\ &\quad +F(2*4*)+F(2**6) \\ &\quad +F(264*)+F(26*6) \\ &\quad +F(2*46)+F(2646) \end{aligned} \quad \left. \begin{array}{l} \text{from dad} \\ \text{from mom} \\ \text{new schemata} \end{array} \right\} \quad (26)$$

Note that "come from" does not mean "actually come from" in this case. If *dad* and *mom* have the same figure x in a locus, we do not have to decide which x is given to *kid*. In the example, *kid* has '6' in locus 1. We can think the crossover occurred between locus 0 and locus 1; *kid*'s 6 comes from *mom*. And also we can think the crossover occurred between locus 1 and locus 2; *kid*'s 6 comes from *dad*.

5.3 Evaluating the building block hypothesis

It is said that genetic algorithms work effectively when the short and low-order schemata combine to form the optima or the near-optima. It is called "building block hypothesis"³⁾. OSD is useful for the evaluation: how the objective function satisfies the building block hypothesis.

Let decompose f by following equations.

$$f = \sum_{i \leq N} fo_i \text{ where } fo_i = \sum_{O(i)=i} f_i. \quad (27)$$

$$f = \sum_{i \leq N} fl_i \text{ where } fl_i = \sum_{L(i)=i} f_i. \quad (28)$$

Let us call fo_i 'order i component of f ' and call fl_i 'length i component of f '.

$$\forall i \neq j [fo_i \perp fo_j]. \quad (29)$$

$$\forall i \neq j [fl_i \perp fl_j]. \quad (30)$$

Therefore,

$$\|f\|^2 = \sum_{i \leq N} \|fo_i\|^2 = \sum_{i \leq N} \|fl_i\|^2. \quad (31)$$

$\|fo_i\|^2$ means the power of order i , and $\|fl_i\|^2$ means the power of length i . Using $\|fo_i\|^2$ and $\|fl_i\|^2$, we can illustrate the order spectrum and the length spectrum as we illustrate the frequency spectrum in the Fourier-transform analysis. And we can say that the building block hypothesis is satisfied when the power concentrate to low-order and short-length.

6. Conclusion

OSD is a useful mathematical tool for the schema analysis of the genetic algorithms. While WST can be used only for the binary gene, OSD can be used also for non-binary gene. As WST has the fast transform algorithms, OSD also has a fast decomposition algorithm. OSD is same as WST when gene is binary. Therefore, OSD is an expansion of WST. OSD is useful for computing the schema's average value, evaluating the inheritance from parents and evaluating how the objective function satisfies the building block hypothesis.

References

- 1) D. E. Goldberg : Genetic Algorithms and Walsh Functions-Part I: A Gentle Introduction ; Complex Systems, Vol. 3, pp. 129~152 (1989)
- 2) D. E. Goldberg : Genetic Algorithms and Walsh Functions-Part II: Deception and Its Analysis ; Complex Systems, Vol. 3, pp. 153~171 (1989)
- 3) D. E. Goldberg : Genetic Algorithms in Search, Optimization & Machine Learning, Addison-Wesley (1989)
- 4) L. Davis : Handbook of Genetic Algorithms, Van Nostrand Reinhold (1990)
- 5) C. L. Bridges and D. E. Goldberg : The Nonuniform Walsh-Schema Transform, Foundations of Genetic Algorithms, Morgan-Kaufmann, pp. 13~22 (1991)

Appendix

Nomenclature

Ω_i is the set of the figures at locus i where 'figure' is an arbitrary symbol except $*$ and $\%$. $0 < |\Omega_i| < \infty$. For example, $\Omega_i = \{0, 1, 2\}$.

Ω is the set of gene : the row of figures. $\Omega = \Omega_0 \times \cdots \times \Omega_{N-1}$ where N is the length of gene. For example, $\Omega = \{0, 1\} \times \{0, 1, 2, 3\} \times \{0, 1, 2\}$.

$*$ is the don't care symbol. $\%$ is the fixed value symbol.

S is the set of schema : the row of symbols which consists of figures and $*$'s. $S = (\Omega_0 + \{*\}) \times \cdots \times (\Omega_{N-1} + \{*\})$. Schema is often treated as a set of genes. For example, $021 \in *21$.

T is the set of schema template : the row of symbols which consists of $*$ and $\%$. $T = \{*, \%\}^N$.

$\leq$ is the order relation in S or T defined by

$$s_0 \cdots s_{N-1} \leq u_0 \cdots u_{N-1} \Leftrightarrow \forall i [u_i = * \Rightarrow s_i = *].$$

For example, $* \% * \leq * \% \%$ and $* 9 * \leq * 9 6$. $s < u$ means that $s \leq u$ and $s \neq u$. ($T, \leq$) is a lattice-ordered set. Therefore $t \wedge u$ (meet of t and u) is defined by

$$(t \wedge u)_i = \begin{cases} \%, & t_i = u_i = \% \\ *, & \text{otherwise} \end{cases}$$

and $t \vee u$ (join of t and u) is defined by

$$(t \vee u)_i = \begin{cases} *, & t_i = u_i = * \\ \%, & \text{otherwise.} \end{cases}$$

For example, $\% * \% \wedge * \% \% = **\%$ and $\% * \% \vee * \% \% = \% \% \%$.

$x \$ t$ ($x \in \Omega$, $t \in T$) is the schema "x masked by t" where

$$(x \$ t)_i = \begin{cases} x_i, & t_i = \% \\ *, & t_i = *. \end{cases}$$

For example, $012 \$ \% * \% = 0 * 2$.

$O(s)$ ($s \in S$) is the order of schema s : the number of figures in s . $O(t)$ ($t \in T$) is the order of schema template t : the number of $\%$ in t . For example, $O(*11) = O(\% * \%) = 2$.

$L(s)$ ($s \in S$) is the length of schema s : the distance between the first figure and last figure in s . $L(t)$ ($t \in T$) is the length of schema template t : the distance between the first $\%$ and the last $\%$ in t . For example, $L(*1*2) = L(\% \% \% *) = 3$. And we define $L(* \cdots *) = 0$.

f is the objective function. $f: \Omega \rightarrow \mathbf{R}$.

V is the function space. $V = \{f | f: \Omega \rightarrow \mathbf{R}\}$.

$\dim(A)$ is the dimension of subspace A .

Σ denotes the sum of subspace.

$$\sum_i A_i = \left\{ \sum_i a_i \mid a_i \in A_i \right\}.$$

$\cdot$ denotes the inner product in V .

$$f \cdot g = \sum_{x \in \Omega} f(x)g(x).$$

$\| \cdot \|$ denotes the Euclidean norm. $\|f\| = (f \cdot f)^{1/2}$.

$a \perp b$ ($a, b \in V$) means that $a \cdot b = 0$.

$a \perp B$ ($a \in V, B \subseteq V$) means that

$$\forall b \in B \ [a \perp b].$$

$A \perp B$ ($A, B \subseteq V$) means that

$$\forall a \in A, b \in B \ [a \perp b].$$

$A^\perp$ ($A \subseteq V$) is the orthocomplement of A .

$$A^\perp = \{v \in V \mid v \perp A\}.$$