# A Crossover Operation for Evolutionary Binary Decision Diagrams

Kai Sugimoto
Nagoya Institute of Technology
Email:ksugimoto@paris.ai.nitech.ac.jp

Tsuyoshi Nakamura
Nagoya Institute of Technology
Email:tnaka@nitech.ac.jp

Masayoshi Kanoh
Chukyo University
Email:mkanoh@sist.chukyo-u.ac.jp

***Abstract*—We propose a crossover operation for multi-terminal decision diagrams (MTBDDs). To survey this crossover operation, we conducted experiments of the evolution of MTBDDs with and without the proposed crossover operation. We confirmed that MTBDDs that have better fitness were obtained in the evolution of MTBDDs with the proposed crossover operation than without it. We also confirmed that MTBDDs that possess the smaller number of vertices were obtained in the evolution of MTBDDs with the proposed crossover operation than without it.**

## I. Introduction

Service robots for domestic tasks and entertainments have recently been developed [1], [2], [3]. These robots are more familiar to us than industrial robots. This trend will continue and more service robots will be developed [4]. Interest is particularly high in humanoid robots because they have human-like bodies and move like humans, and function in our daily environments. However, such robots that possess complex mechanisms are difficult to control and require complex mechanical computation. To avert such computation, Kanoh et al. [5] proposed to control robots by using multi-terminal binary decision diagrams (MTBDDs) and to obtain the graph structure of MTBDDs by the evolution of MTBDDs with genetic programming. The "evolution of MTBDDs" indicates to modify graph structure of MTBDDs by using evolutionary computation. In this research, robot joint angles are inputted into an MTBDD to control robots. However the MTBDD variables take only binary values, so multiple variables are needed to represent a single robot joint angle. This creates a large number of variables. To solve this problem, Sakai et al. [6] used multi-valued decision diagrams (MDDs) to control robots. Its variables take several values, so only a single variable is needed to represent a single robot joint angle. They argued that MDDs are more effective than MTBDDs in discovering important variables from decision diagrams. However, these researches used genetic programming without a crossover operation to evolve decision diagrams. Generally, genetic programming is used in tree structures. In a tree structure, subtrees can be created easily, so a crossover operation by sub-trees swapping can be defined. In decision diagrams, sub-diagrams swapping is difficult because decision diagrams are the graph structures that possess closed paths. A crossover operation is important for the maintenance of diversity in genetic programming. If a crossover operation is used in the evolution of decision diagrams, we expect to obtain the diagrams with better fitness. APPLY crossover, which is a crossover operation for decision diagrams, has been proposed [7]. However, this operation can only be applied to decision diagrams that possess the same variable orders. To solve this problem, Flexible APLLY crossover [8], which can be applied to decision diagrams that possess different variable orders, has been proposed by extending APPLY crossover. However, sub-structures of diagrams of the individuals generated by these crossover operations might be lost because the input-output relation of the individuals is generated stochastically. In this paper, we propose a crossover operation for MTBDDs that possess the different variable orders. To confirm effectiveness of this crossover operation, we conducted the experiments on the evolution of MTBDDs by using genetic programming with and without the proposed crossover operation and compared the results.

## II. Evolutionary Binary Decision Diagrams

### A. Binary Decision Diagrams

Binary decision diagrams (BDDs) [9] are data structures that are compact representations of Boolean functions. They consist of two types of vertices (i.e., non-terminal vertices and terminal vertices) and two types of edges (i.e., 0-edge and 1-edge). Edges connect upper non-terminal vertices and lower non-terminal vertices or upper non-terminal vertices and lower terminal vertices. Each non-terminal vertex possesses one 0-edge and one 1-edge. If the value of the variable assigned to the non-terminal vertex is 0, the 0-edge is followed; when the value is 1, the 1-edge is followed. A terminal vertex is assigned a value of 0 or 1. The processing of the variable proceeds from the non-terminal vertex at the root of the diagram in order until a terminal vertex is reached.

An example a BDD is shown in Fig. 1, where non-terminal vertices are represented by circles and terminal vertices are represented by squares. In this BDD, three variables $(x_1, x_2, x_3)$ can be processed. For example, when $(x_1, x_2, x_3) = (0, 1, 1)$ is input, first the 0-edge is followed from the non-terminal vertex $x_1$ located at depth 0. Then, the 1-edge is followed from non-terminal vertex $x_3$ at depth 2 to arrive at the output of 1.

### B. Multi-terminal binary decision diagrams

Multi-terminal binary decision diagrams (MTBDDs) extend BDDs to have multiple output values. That is, MTBDDs allow the terminal vertices to have multiple values other than 0 or 1.

In the example MTBDD shown in Fig. 2, three variables $(x_1, x_2, x_3)$ can be processed. For example, when

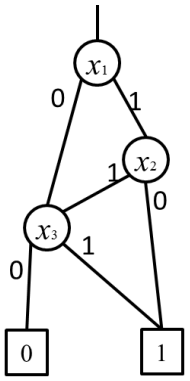



Fig. 1. Example of BDD

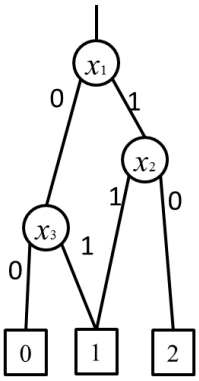



Fig. 2. Example of MTBDD

$(x_1, x_2, x_3) = (0, 1, 1)$ is input, first the 0-edge is followed from the non-terminal vertex $x_1$ located at depth 0. Next, the 1-edge is followed from non-terminal vertex $x_3$ at depth 2 to arrive at the output of 1.

### C. Evolutionary Binary Decision Diagrams

In this paper, we use genetic programming in the evolution of MTBDDs. The procedure is as follows.

i) Initialize generation $g \leftarrow 0$.
ii) Generate initial population whose elements are MTBDDs $U = \{u_1, u_2, ..., u_n\}$, where $n$ is the number of individuals.
iii) Compute the fitness of the individuals in $U$.
iv) Select the top $k$ individuals from $U$, and let them be the set $P = \{p_1, p_2, ..., p_k\}$.
v) $Q = U \setminus P$, and perform genetic operations to $Q$. The set after this procedure is $Q'$.
vi) Generate the next-generation set $U = P \cup Q'$.
vii) $t \leftarrow t + 1$return to step iii).

Moriwaki [10] and Kanoh [5] proposed the three operations as genetic operations for MTBDDs. (i.e., insertion, mutation, deletions).

*a) Insertion (Fig. 3):* $Insertion$ adds a new nonterminal vertex above a randomly selected edge. One of the edges of the added non-terminal vertex remains connected to the vertex to which it was connected prior to the addition. The other edges connect to any lower non-terminal or vertices.

*b) Mutation (Fig. 4):* $Mutation$ changes the destination of one randomly selected edge of a non-terminal vertex to a randomly selected subordinate non-terminal or terminal vertices.

*c) Deletion (Fig. 5):* $Deletion$ deletes a randomly selected non-terminal vertex. The edges that are connected to the delete vertex are set to connect to the vertices to which the edges of the deleted vertex are pointed.

We believe all these genetic operations are classed as mutations in the general genetic operation because they only add or delete a single vertex or change the destination of a single edge. It is possible that the characteristics the parents possess are inherited by only mutation, but this is inefficient. In this paper, the proposed crossover operation is introduced in the evolution of MTBDDs.

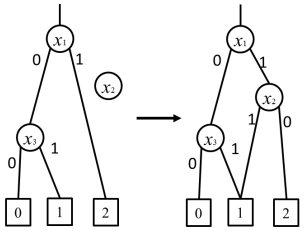



Fig. 3. insertion

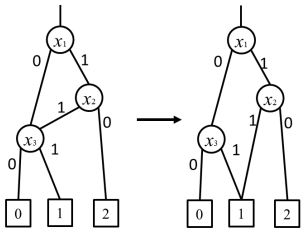



Fig. 4. mutation

## III. Crossover Operation

A crossover operation generates a new individual by combining two diagrams. In this paper, we propose a crossover operation that is based on sub-diagrams swapping by referring to the crossover operation for tree structures.

Figure 6 shows the algorithm of the proposed crossover operation.

With this algorithm, the crossover operation by sub-diagrams swapping is possible in graph structures that possess closed paths. We explain the algorithm by using Figure 7. In this figure, $ParentA$ is $d_1$ and $ParentB$ is $d_2$. First, create set V (lines 4-8 in Fig. 6 ). The element of V is a combination of the two non-terminal vertices in which one is the vertex in $ParentA$ and the other is the vertex in $ParentB$. The suffixes of the variables of both vertices are the same. When there is no non-terminal vertices that possess the same suffix of variables; $V = \phi$, the algorithm ends (lines 9-11 in Fig. 6 ). Next, in $ParentA$, delete all non-terminal vertices that can be reached from $v1$ (line 14 in Fig. 6 and Fig. 7(c)) then, in the combination of non-terminal vertices $(v1, v2)$ selected randomly from $V$, change the destination of edge $e$ of which destination is $v1$, into $v2$ (line 15 in Fig. 6 and Fig. 7(d)). At this time, in $ParentA$ randomly change the destinations of edges that do not possess any destination (line 16 in Fig. 6 and 1-edge of $x_2$ in Fig. 7(e)). Finally, reduce $ParentA$ and output $ParentA$ as a result of the crossover operation (lines 17-18 in Fig. 6 and Fig. 7(f)).

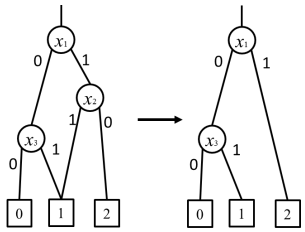



Fig. 5. deletion

**Crossover Operation**
Input $Q$ : a set of BDDs.
Output $ans$ : a BDD. (If crossover is possible)
: $false$. (If crossover is impossible)

1. **begin**
2. Select two BDDs $d_1, d_2 \in Q$ randomly;
3. Equalize variable orders in $d_1, d_2$;
4. $V \leftarrow \phi$;
5. **for each** $v_{x_i}^{(d_1)} \in d_1$
   % $v_{x_i}^{(d_1)}$is a non-terminal vertex that possesses $x_i$
6. **for each** $v_{x_j}^{(d_2)} \in d_2$
   % $v_{x_j}^{(d_2)}$is a non-terminal vertex that possesses $x_j$
7. **if** $i = j$
8. **then** the edge whose destination is $v_{x_i}^{(d_1)}$ is $e$, $V \leftarrow V \cup \{(e, v_{x_i}^{(d_1)}, v_{x_j}^{(d_2)})\}$;
9. **if** $V = \phi$ **then begin**
10. $ans \leftarrow false$;
11. **go to** 19.
12. **end**
13. Select $(e, v_{x_k}^{(d_1)}, v_{x_k}^{(d_2)}) \in V$ randomly;
14. Delete all non-terminal vertices that can be reached from $v_{x_k}^{(d_1)}$;
15. Change the destination of $e$ into $v_{x_k}^{(d_2)}$;
16. In $d_1$, randomly connect the edges that possess no destination to the lower vertex than the vertex that possesses the edges;
17. Reduce $d_1$;
18. $ans \leftarrow d_1$;
19. **end.**

Fig. 6. Crossover operation algorithm

We explain the method of equalizing different variable orders of two BDDs at line 3 in Fig. 6. Swapping the order of two adjacent variables in a BDD affect only non-terminal vertices assigned these variables; all other vertices remain unchanged [11]. This is obvious by considering the combination of inputs-outputs represented by two variables. Let $x^{(n)}$ be a variable placed at depth n, and $f(a, b)$ be an output obtained by inputting $x^{(n)} = a \in \{0, 1\}$ and $x^{(n+1)} = b \in \{0, 1\}$ to $v_{x^{(n)}}$. Figure 8 shows that the same function can be represented after variable order changes. This property is the same in MTBDDs. Swapping the order of two adjacent variables by bubble sort enables to creation of a BDD having the desired order (Fig. 9).

## IV. Evolution Simulation Experiments

### A. Experimental outline

To confirm the effectiveness of the proposed crossover operation, we conducted simulation experiments to obtain motion representation of a humanoid robot standing up from being seated in a chair by using MTBDDs. In this paper, we focused on the evaluation on proposed crossover operation. We compared the results of the evolution of MTBDD with and without the proposed crossover operation. For the humanoid robot, we used the HOAP-1 (Humanoid for Open Architecture Platform) produced by Fujitsu Automation. The robot is 48 cm tall, weighs 6 kg, has 20 degrees of freedom (DOFs), and has four pressure sensors, each on the soles its feet. Additionally, angular rate and acceleration sensors are mounted on its chest.

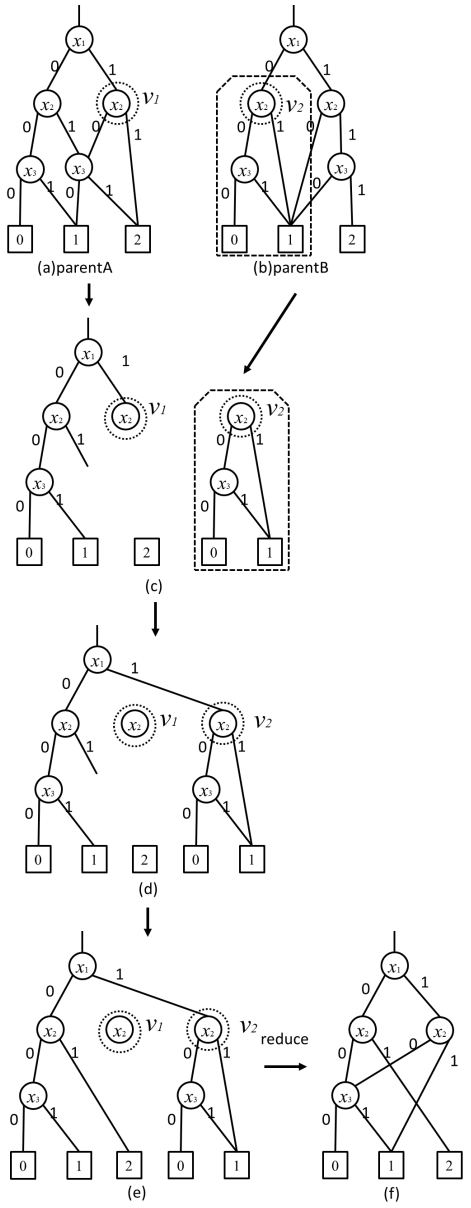



Fig. 7. Example of crossover

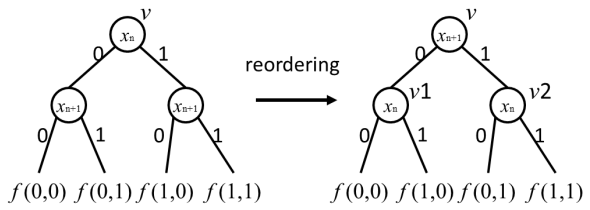



Fig. 8. Overview of reordering

Figure 10 show HOAP-1. Figure 11 illustrates the joint angles we used. The humanoid robot by usage of a MTBDD obtains the following data from the sensors:

$$\boldsymbol{x} = (x_0^W, x_1^W, x_0^K, x_1^K, x_0^A, x_1^A, x_0^B, x_1^B) \qquad (1)$$

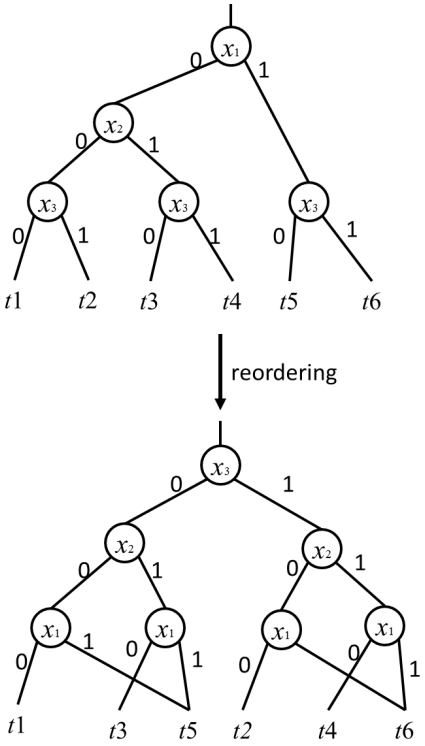



Fig. 9. Example of reordering

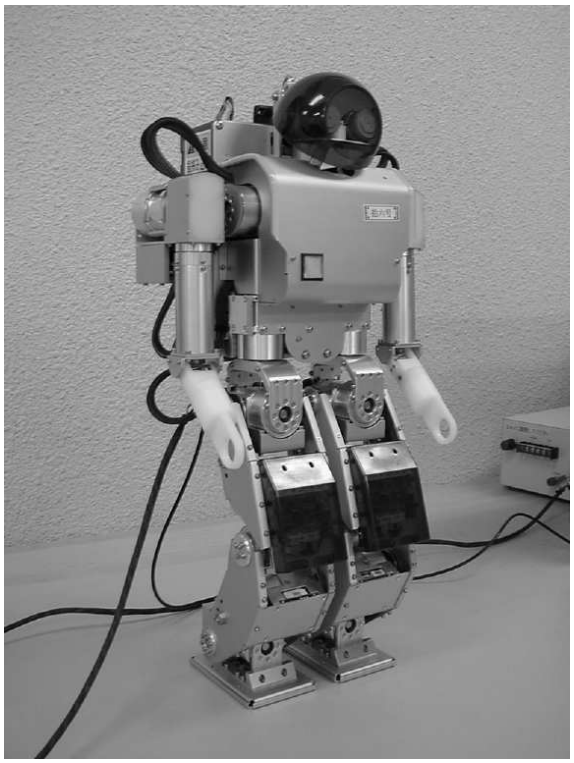


Fig. 10. Overview of HOAP-1 (Humanoid for open architecture platform)

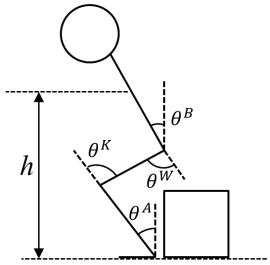



Fig. 11. Robot and joint angles

where $x_i \in \{0, 1\}$ and $x_i^W$, $x_i^K$, and $x_i^A$ are variables that represent the angles of the waist, knee, and ankle, respectively, and $x_i^B$ represents the pitch of the body. These variables values are determined from the current joint angle values listed in Table I. In this table, $\theta_{\min}$ and $\theta_{\max}$ are limiting values of the motor range of motions, which are defined as $\theta^W_{\min} = -80.0\,[\mathrm{deg}]$, $\theta^W_{\max} = 70.0\,[\mathrm{deg}]$, $\theta^K_{\min} = 0.0\,[\mathrm{deg}]$, $\theta^K_{\max} = 120.0\,[\mathrm{deg}]$, $\theta^A_{\min} = -60.0\,[\mathrm{deg}]$, $\theta^A_{\max} = 60.0\,[\mathrm{deg}]$, $\theta^B_{\min} = -60.0\,[\mathrm{deg}]$, and $\theta^B_{\max} = 60.0\,[\mathrm{deg}]$. The variables of the MTBDDs are placed from depths 0 to 7 because the total number of variables is 8. The robot action outputs are listed in Table II, where the plus signs '+' signify that the joint is moved at $20.0\,[\mathrm{deg/sec}]$ and the minus signs '−' signify that the joint is moved at $-20.0\,[\mathrm{deg/sec}]$.

The robot control time interval was set as $\Delta t = 0.01[\mathrm{s}]$. One trial ended when the robot fell down or time exceeded 10 s. The fitness was calculated the sum of the values of the robot's chest position $h(t)\,[\mathrm{m}]$ at all times $t$ during t atrial (i.e., $Fitness = \Sigma_t h(t)$)). To simulate the evolution of robot motion, we used the open dynamics engine [12].

The probabilities of genetic operations, in the evolution of MTBDD with the proposed crossover operation, are $crossover = 0.5$, $insertion = 0.3$, $mutation = 0.15$, and $deletion = 0.05$, and the probabilities of genetic operations in the evolution of MTBDD without the proposed crossover operation are $insertion = 0.6$, $mutation = 0.3$, $deletion = 0.1$. Both ratios of the probabilities of genetic operations except $crossover$ are equal.

In the proposed crossover operation, one variable order is selected from the variable order of two parent MTBDDs and the variable order of them are equalized to this variable order before sub-diagrams swapping. In these experiments, this variable order is that the sum of the number of vertices in two parent MTBDDs is smaller when the variable orders of them are equalized to this variable order than when the variable orders of them are equalized to the other variable order. The reason for this is that we can easily recognize the rule of MTBDDs that can be represented with the small number of vertices.

The MTBDDs in the initial population are generated using 30 insertion operations because the proposed crossover operation greatly affects the MTBDDs that possess a sufficient number of vertices.

### *B. Results and discussion*

Fig. 12 shows one example of the acquired motions; standing up. Figure 13 shows the fluctuation in fitness. We can see that the results of the evolution of MTBDDs with the proposed crossover operation were better fitness than those without the proposed crossover operation. In early generations,

TABLE I. CORRESPONDENCE BETWEEN ANGLE DATA AND VARIABLE VALUES

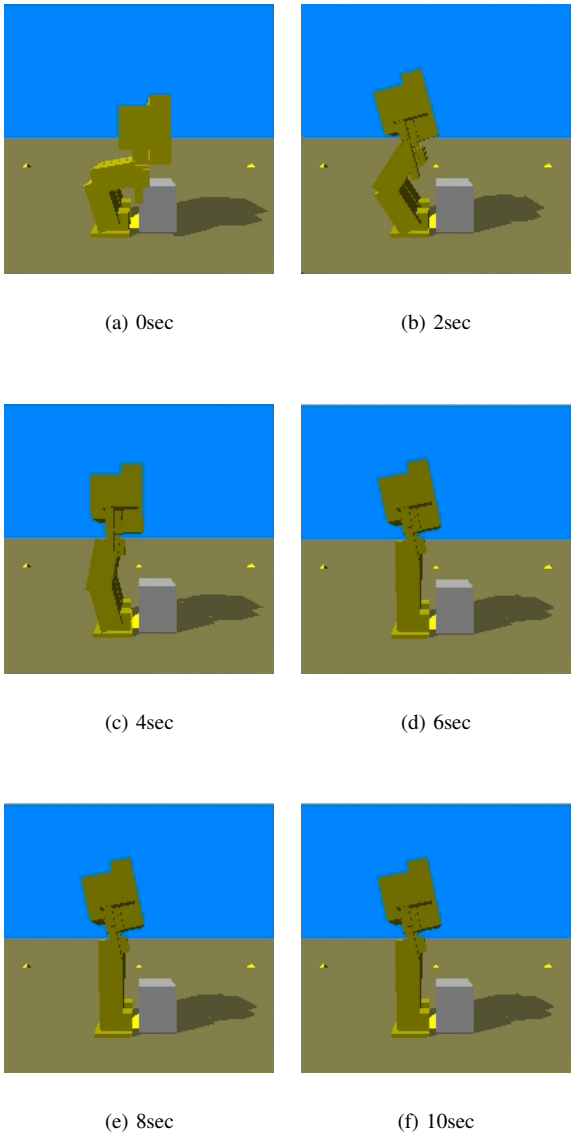
| Sensed degree | $(x_0, x_1)$ |
|---|---|
| $\theta_{\min} \leq \theta < \frac{1}{4}\theta_{\max} + \frac{3}{4}\theta_{\min}$ | $(0,0)$ |
| $\frac{1}{4}\theta_{\max} + \frac{3}{4}\theta_{\min} \leq \theta < \frac{1}{2}\theta_{\max} + \frac{1}{2}\theta_{\min}$ | $(0,1)$ |
| $\frac{1}{2}\theta_{\max} + \frac{1}{2}\theta_{\min} \leq \theta < \frac{3}{4}\theta_{\max} + \frac{1}{4}\theta_{\min}$ | $(1,0)$ |
| $\frac{3}{4}\theta_{\max} + \frac{1}{4}\theta_{\min} \leq \theta < \theta_{\max}$ | $(1,1)$ |

TABLE II. TYPES OF BEHAVIOR

| Terminal vertex | Motor output | | |
|---|---|---|---|
| | Waist | Knee | Ankle |
| $a_0$ | + | + | + |
| $a_1$ | + | + | − |
| $a_2$ | + | + | 0 |
| $a_3$ | + | − | + |
| $a_4$ | + | − | − |
| $a_5$ | + | − | 0 |
| $a_6$ | + | 0 | + |
| $a_7$ | + | 0 | − |
| $a_8$ | + | 0 | 0 |
| $a_9$ | − | + | + |
| $a_{10}$ | − | + | − |
| $a_{11}$ | − | + | 0 |
| $a_{12}$ | − | − | + |
| $a_{13}$ | − | − | − |
| $a_{14}$ | − | − | 0 |
| $a_{15}$ | − | 0 | + |
| $a_{16}$ | − | 0 | − |
| $a_{17}$ | − | 0 | 0 |
| $a_{18}$ | 0 | + | + |
| $a_{19}$ | 0 | + | − |
| $a_{20}$ | 0 | + | 0 |
| $a_{21}$ | 0 | − | + |
| $a_{22}$ | 0 | − | − |
| $a_{23}$ | 0 | − | 0 |
| $a_{24}$ | 0 | 0 | + |
| $a_{25}$ | 0 | 0 | − |
| $a_{26}$ | 0 | 0 | 0 |

the increase in fitness in evolution of MTBDDs with the proposed crossover operation was faster. We consider that the diversity of individuals is maintained by the proposed crossover operation in early generations, which affects the faster increase in fitness in early generations.

Fig. 14 shows the fluctuation in the number of non-terminal vertices. We can see that the results of the evolution of MTBDD with the proposed crossover operation were the rapidly increase in the number of non-terminal vertices in early generations. We consider that irreducible MTBDDs that possess a large number of vertices are generated by applying the proposed crossover operation to MTBDDs, in which the graph substructures are very different, and that it is possible that MTBDDs that possess a large number of vertices have better fitness because they can represent various input-output relations. We consider that this reason for accelerating the evolution of MTBDD with the proposed crossover is this. It should be noted that quasi-optimum MTBDDs that possess a smaller number of vertices are obtained. We consider that the important MTBDD substructures in the control of robot are obtained then, as the features are optimized, individuals with better fitness and possessing a small number of vertices are obtained. The MTBDDs possessing small number of vertices are easy to recognize, because the rule represented by the MTBDDs is simple. MTBDDs that represent a simple rule for controlling robots are effective when people know that rule. We



(a) 0sec (b) 2sec

(c) 4sec (d) 6sec

(e) 8sec (f) 10sec

Fig. 12. Evolution results for MTBDD (500th generation)

conclude that the proposed crossover operation is an effective in promoting recognition of control rue of robot.

## V. CONCLUSION

In this paper, we proposed a crossover operation for MTBDDs. To survey the proposed crossover operation, we conducted experiments on obtaining motion representation for a humanoid robot standing up from being seated in a chair in the evolution of MTBDD with the crossover operation. We conducted the same experiment in the evolution of MTBDD without the proposed crossover operation. We confirm that MTBDDs which have better fitness were obtained in the evolution of MTBDD with the proposed crossover operation than without it. We also confirm that MTBDDs that possess a smaller number of vertices were obtained in the evolution of MTBDD with the proposed crossover operation without it.

For future works, we will compare Flexible APLLY

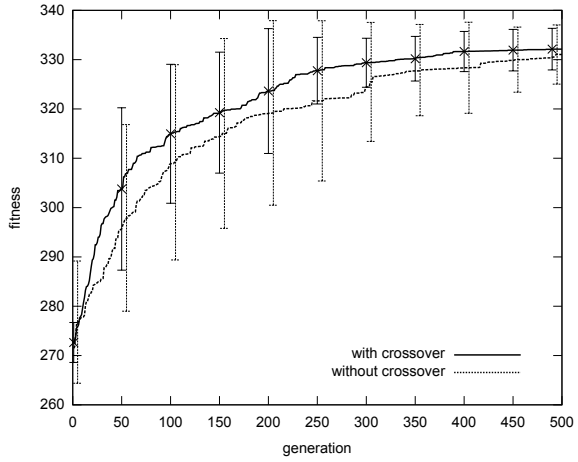



Fig. 13. Relation of generations and fitness (average for 20 times)

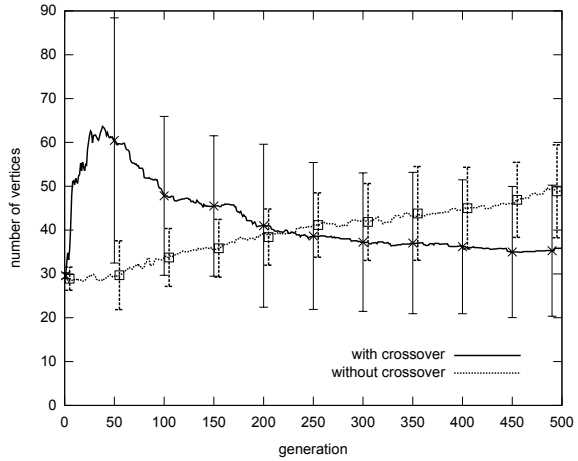



Fig. 14. Relation of generations and number of vertices (average for 20 times)

crossover and the proposed crossover operation and evaluate each property of these operations. We will also modify this operation for MDDs and compare the results to Sakai et al.'s research [6], in which they argue that the evolution of MDDs has the problem in which the fitness of the obtained individuals is worse than the evolution of MTBDDs. We will confirm whether it is possible to solve this problem by using the proposed crossover operation.

## REFERENCES

[1] J. Sung, R. E. Grinter, H. I. Christensen, and L. Guo, "Housewives or Technophiles?: Understanding Domestic Robot Owners," in *HRI '08: Proceedings of the 3rd ACM/IEEE international conference on Human robot interaction*. New York, NY, USA: ACM, 2008, pp. 129–136.

[2] J. Forlizzi and C. DiSalvo, "Service robots in the domestic environment: a study of the roomba vacuum in the home," in *HRI '06: Proceedings of the 1st ACM SIGCHI/SIGART conference on Human-robot interaction*. New York, NY, USA: ACM, 2006, pp. 258–265.

[3] P. Saulnier, E. Sharlin, and S. Greenberg, "Using bio-electrical signals to influence the social behaviours of domesticated robots," in *Proceedings of the 4th ACM/IEEE international conference on Human robot interaction*. ACM, 2009, pp. 263–264.

[4] International Federation of Robotics, "Service robot statistics," (2013, Dec.). [Online]. Available: http://www.ifr.org/service-robots/statistics/

[5] M. Kanoh and H. Itoh, "A New Dynamic Insertion Operation for n-BDD -Applying to Obtaining Robot Controller-," *Journal of Japan Society for Fuzzy Theory and Intelligent Informatics*, vol. 20, no. 6, pp. 909–920, 2008, in Japanese.

[6] M. Sakai, M. Kanoh, and T. Nakamura, "Evolutionary Multivalued Decision Diagrams for Obtaining Motion Representation of Humanoid Robots." *IEEE Transactions on Systems, Man, and Cybernetics, Part C*, vol. 42, no. 5, pp. 653–663, 2012.

[7] A. Mutoh, S. Oono, K. Moriwaki, N. Inuzuka, and H. Itoh, "A food chain model using APPLY crossover of n-output Binary Decision Diagram," *The transactions of the Institute of Electrical Engineers of Japan. C, A publication of Electronics, Information and System Society*, vol. 121, no. 2, pp. 423–429, 2001, in Japanese.

[8] A. Mutoh, T. Sawada, S. Kato, and H. Itoh, "A Model of Biological Differentiation using Flexible APPLY crossover of n-output Binary Decision Diagram," *journal of Japan Society for Fuzzy Theory and Intelligent Informatics*, vol. 21, no. 2, pp. 236–246, 2009, in Japanese.

[9] R. E. Bryant, "Graph-Based Algorithms for Boolean Function Manipulation," *IEEE Transactions on Computers*, vol. 35, no. 8, pp. 677–691, 1986.

[10] K. Moriwaki, D. Yokoi, N. Inuzuka, and H. Itoh, "Evolution of n-BDD Using Genetic Programming Method: a Food Chain Simulation of Evolutional Agents," *Journal of Japanese Society for Artificial Intelligence*, vol. 14, no. 3, pp. 477–484, 1999, in Japanese.

[11] R. Rudell, "Dynamic variable ordering for ordered binary decision diagrams," in *Proceedings of the 1993 IEEE/ACM international conference on Computer-aided design*. IEEE Computer Society Press, 1993, pp. 42–47.

[12] R. Smith, (2009, Jan.) Open Dynamics Engine. [Online]. Available: http://www.ode.org/