



논문 스터디 5주차

Oct 1, 2021

Genetic Algorithm

UNIST brAIns 유인재

CONTENTS

1. Intro.

2. Basic Concepts

3. Theory & Hypothesis

4. Structure Modifications

5. Applications

6. Conclusion

Intro.

Intro.

Genetic Algorithms : Concepts and Applications (1996), IEEE

K. F. Man, K. S. Tang, and S. Kwong, Member, IEEE

Intro.

Basic Concepts

Theory / Hypothesis

Structure Modification

Applications

Conclusions

Genetic Algorithms: Concepts and Applications

K. F. Man, Member, IEEE, K. S. Tang, and S. Kwong, Member, IEEE

Abstract—This paper introduces genetic algorithms (GA) as a complete entity, in which knowledge of this emerging technology can be integrated together to form the framework of a design tool for industrial engineers. An attempt has also been made to explain “why” and “when” GA should be used as an optimization tool.

I. INTRODUCTION

THE USE of genetic algorithms (GA) for problem solving is not new. The pioneering work of J. H. Holland in the 1970's proved to be a significant contribution for scientific and engineering applications. Since then, the output of research work in this field has grown exponentially although the contributions have been, and are largely initiated, from academic institutions world-wide. It is only very recently that we have been able to acquire some material that comes from industry. The concept of this is somehow not clearly understood. However, the obvious obstacle that may drive engineers away from using GA is the difficulty of speeding up the computational process, as well as the intrinsic nature of randomness that leads to a problem of performance assurance.

Nevertheless, GA development has now reached a stage of maturity, thanks to the effort made in the last few years by academics and engineers all over the world. It has blossomed rapidly due to the easy availability of low-cost but fast-speed small computers. Those problems once considered to be “hard” or even “impossible,” in the past are no longer a problem as far as computation is concerned. Therefore, complex and conflicting problems that require simultaneous solutions, which in the past were considered deadlocked problems, can now be obtained with GA.

Furthermore, the GA is not considered a mathematically guided algorithm. The optima obtained is evolved from generation to generation without stringent mathematical formulation such as the traditional gradient-type of optimizing procedure. In fact, GA is much different in that context. It is merely a stochastic, discrete event and a nonlinear process. The obtained optima is an end product containing the best elements of previous generations where the attributes of a stronger individual tend to be carried forward into the following generation. The rule of the game is “survival of the fittest will win.”

In this sphere, there is an endless supply of literature describing the use of GA. The sheer number of references quoted in this paper is an apt indicator of the extensive work being done in this domain. This does not include the index finding from [1]. The technical knowledge of “what” GA is and “how” it works are well reported. This paper tries not to cover the same ground. Rather, there is room for the introduction of

Manuscript received September 21, 1995; revised November 26, 1995. The authors are with the City University of Hong Kong, Hong Kong. Publisher Item Identifier S 0278-0046/96/03295-9.

GA as a complete entity, in which knowledge of this emerging technology can be integrated together to form the framework of a design tool for industrial engineers. Moreover, a brave attempt has also been made to explain “why” and “when” we should use GA as an optimization tool. It is anticipated that there is sufficient materials being generated in this paper to support this claim.

This paper starts by giving a simple example of GA, as described in Section II, in which the basic framework of GA is outlined. This example forms the cornerstone to the architecture of this paper. For the benefit of newcomers to this particular field, the essential schema theory and building block hypothesis of genetic algorithms are briefly given in Section III. What makes GA work and how does it improve its evolution are the essence of GA. There are a number of variations used to achieve these tasks, and each task has its own merit. In Section IV, a range of structural modifications for GA in order to improve its performance are thus recommended.

Since so much has already been published about what can be done with GA, a short list of items cataloguing the advantages of using GA is given in Section V. In Section VI, an account of what GA “cannot do” is given. The well-known phenomena of deception and genetic drift are described. In addition, the problems concerning the real time and adaptiveness of GA are also reported.

As this paper is targeted at a specific audience, the collection of practical systems being implemented are introduced in Section VII, whereas Section VIII outlines the possibility of integrating GA into emerging technologies such as neural networks and fuzzy systems. Finally, the conclusions reached in Section XI and recommendations for future works are also given.

II. BASIC CONCEPTS OF GENETIC ALGORITHMS

The basic principles of GA were first proposed by Holland [66]. Thereafter, a series of literature [33], [52], [89] and reports [10], [11], [102], [118] became available. GA is inspired by the mechanism of natural selection, a biological process in which stronger individuals are likely to be the winners in a competing environment. Here, GA uses a direct analogy of such natural evolution. It presumes that the potential solution of a problem is an individual and can be represented by a set of parameters. These parameters are regarded as the genes of a chromosome and can be structured by a string of values in binary form. A positive value, generally known as fitness value, is used to reflect the degree of “goodness” of the chromosome for solving the problem, and this value is closely related to its objective value.

[71] C. Z. Janikow and Z. Michalewicz, “An experimental comparison of binary and floating point representations in genetic algorithms,” in *Proc. 4th Int. Conf. Genetic Algorithms*, July 1991, pp. 31–36.

[72] A. H. Jones and P. B. De Moura Oliveira, “Genetic auto-tuning of PID controllers,” in *1st IEEE/IEEE Int. Conf. GA's in Engineering Systems: Innovations and Applications*, Sheffield, U.K., 1995, pp. 141–145.

[73] C. L. Karr, “Genetic algorithms for fuzzy controllers,” *AI Expert*, vol. 6, no. 2, pp. 26–33, 1991.

[74] C. L. Karr and E. J. Gentry, “Fuzzy control of pH using genetic algorithms,” *IEEE Trans. Fuzzy Syst.*, vol. 1, pp. 46–53, 1993.

[75] A. J. Katz and P. R. Thirif, “Generating image filters for target recognition by genetic learning,” *IEEE Trans. Pattern Anal. Machine Intell.*, vol. 16, pp. 906–910, Sept. 1994.

[76] Lj. Kokarev, K. S. Halle, K. Eckert, U. Parlitz, and L. O. Chua, “Experimental demonstration of secure communications via chaos synchronization,” *Int. J. Bifurcation, Chaos*, vol. 2, no. 3, pp. 709–713, 1992.

[77] J. Koza, “Evolution and co-evolution of computer programs to control independently-acting agents,” in *Animals to Animals*, J. A. Ineger and S. W. Wilson, Eds. Cambridge, MA: MIT Press/Bradford Books, 1991.

[78] B. Kruger, P. Schwendler, and O. Vornberger, “Parallel genetic packing on transputers,” in *Parallel Genetic Algorithms: Theory and Applications*. Amsterdam: IOS Press, 1993, pp. 151–185.

[79] S. Kwong, A. C. L. Ng, and K. F. Man, “Improving local search in genetic algorithms for numerical global optimization using modified GRID-point search technique,” in *1st IEEE/IEEE Int. Conf. on GA's in Engineering Systems: Innovations and Applications*, Sheffield, U.K., 1995, pp. 419–423.

[80] W. B. Langdon, (1995). *Scheduling Planned Maintenance of the (U.K.) National Grid*. [Online]. Available: cs.ucl.ac.uk/genetic/papers/grpib_aish-95.ps.

[81] G. Mackie, D. A. Sovic, and G. A. Walters, “Application of genetic algorithms to pump scheduling for water supply,” in *1st IEEE/IEEE Int. Conf. on GA's in Engineering Systems: Innovations and Applications*, Sheffield, U.K., 1995, pp. 403–405.

[82] S. W. Mahfood, “Crowding and preselection revisited,” Dept. Comput. Sci., Univ. Illinois at Urbana-Champaign, IIRGAL Rep. 92004, Apr. 1992.

[83] ———, “Population sizing for sharing methods,” Dept. Comput. Sci., Univ. Illinois at Urbana-Champaign, IIRGAL Rep. 94005, Aug. 1994.

[84] K. L. Mak and Y. S. Wong, “Design of integrated production-inventory-distribution systems using genetic algorithm,” in *1st IEEE/IEEE Int. Conf. on GA's in Engineering Systems: Innovations and Applications*, Sheffield, U.K., 1995, pp. 454–460.

[85] B. Mandrick and P. Spiesens, “Fine-grained parallel genetic algorithms,” in *Proc. 3rd Int. Conf. Genetic Algorithms*, 1989, pp. 428–433.

[86] V. Maniezco, “Genetic evolution of the topology and weight distribution of neural networks,” *IEEE Trans. Neural Networks*, vol. 5, pp. 39–53, Jan. 1994.

[87] *MATLAB User's Guide*. The MathWorks, 1991.

[88] D. McFarlane and K. Glover, “Robust controller design using normalized coprime factor plant descriptions,” in *Information and Control Sciences*. Berlin: Springer-Verlag, 1990.

[89] Z. Michalewicz, *Genetic Algorithms + Data Structures = Evolution Program*, 2nd Ed. Berlin: Springer-Verlag, 1994.

[90] N. H. Moon, A. S. I. Znober, and P. J. Harley, “Sliding mode control design using genetic algorithms,” in *1st IEEE/IEEE Int. Conf. GA's in Engineering Systems: Innovations and Applications*, Sheffield, U.K., 1995, pp. 238–244.

[91] H. Muhlenbein, “Parallel genetic algorithms, population genetics, and combinatorial optimization,” in *Parallelism, Learning, Evolution*. Berlin: Springer-Verlag, 1989, pp. 398–406.

[92] R. Nakano, “Conventional genetic algorithms for job-shop problems,” in *Proc. 4th Int. Conf. Genetic Algorithms*, 1991, pp. 474–479.

[93] R. Nambiar and P. Mars, “Adaptive IIR filtering using natural algorithms,” in *Workshop on Natural Algorithms in Signal Processing*, Chelmsford, Essex, Nov. 1993, pp. 201–2010.

[94] S. Ohayashi, “Genetic algorithm for aerodynamic inverse optimization problems,” in *1st IEEE/IEEE Int. Conf. GA's in Engineering Systems: Innovations and Applications*, Sheffield, U.K., 1995, pp. 7–12.

[95] D. Park, A. Kandil, and G. Langholz, “Genetic-based new fuzzy reasoning models with application to fuzzy control,” *IEEE Trans. Syst., Man, Cybern.*, vol. 24, pp. 39–47, 1994.

[96] G. Roth and M. D. Levine, “Geometric primitive extraction using a genetic algorithm,” *IEEE Trans. Pattern Anal. Machine Intell.*, vol. 16, pp. 901–905, Sept. 1994.

[97] J. D. Schaffer, D. Whitley, and L. J. Eshelman, “Combinations of genetic algorithms and neural networks: A survey of the state of the art,” in *Proc. CGAANN-92 Int. Workshops on Combination of Genetic Algorithms and Neural Networks*, Baltimore, MD, June 6, 1992.

[98] V. Schnecke and O. Vornberger, “Genetic design of VLSI-layouts,” in *1st IEEE/IEEE Int. Conf. GA's in Engineering Systems: Innovations and Applications*, Sheffield, U.K., 1995, pp. 430–435.

[99] K. G. Shin and P. Ramanathan, “Real-time computing: A new discipline of computer science and engineering,” in *Proc. IEEE*, vol. 82, no. 1, Jan. 1994.

[100] J. J. Shynk, “Adaptive IIR filtering,” *IEEE ASSP Mag.*, pp. 4–21, Apr. 1989.

[101] W. M. Spears and K. DeJong, “An analysis of multi-point crossover,” in *Foundations of Genetic Algorithms*, G. J. E. Rawlins, Ed. 1991, pp. 301–315.

[102] M. Srinivas and L. M. Patnaik, “Genetic algorithms: A survey,” *Computer*, pp. 17–26, June 1994.

[103] K. Suzuki and Y. Kakazu, “An approach to the analysis of the basins of the associative memory model using genetic algorithms,” in *Proc. 4th Int. Conf. Genetic Algorithms*, 1991, pp. 539–546.

[104] G. Syswenda, “Uniform crossover in genetic algorithms,” in *Proc. 3rd Int. Conf. Genetic Algorithms*, 1989, pp. 2–9.

[105] ———, “Schedule optimization using genetic algorithms,” in *Handbook of Genetic Algorithms*, pp. 332–349, 1991.

[106] H. Tamaki and Y. Nishikawa, “A parallelized genetic algorithm based on a neighborhood model and its application to job shop scheduling,” *Parallel Problem Solving from Nature*, 2, pp. 575–582, 1992.

[107] K. S. Tang, K. F. Man, and C. Y. Chan, “Fuzzy control of water pressure using genetic algorithm,” in *Proc. IFAC Workshop on Safety, Reliability, and Applications of Emerging Intelligent Control Technologies*, Hong Kong, Dec. 1994, pp. 15–20.

[108] K. S. Tang, K. F. Man, and S. Kwong, “GA approach to time-variant delay estimation,” in *Int. Conf. Control, Information*, Hong Kong, June 5–9, 1995.

[109] K. S. Tang, C. Y. Chan, K. F. Man, and S. Kwong, “Genetic structure for an topology and weights optimization,” in *1st IEEE/IEEE Int. Conf. GA's in Engineering Systems: Innovations and Applications*, Sheffield, U.K., 1995, pp. 250–255.

[110] K. S. Tang, K. F. Man, and D. W. Gu, “Structured genetic algorithm for robust H_{∞} control system design,” *IEEE Trans. Ind. Electron.*, this issue, pp. 575–582.

[111] Y. C. Tang, *Tolkien Reference Manual*. Dept. Comput. Sci., Chinese Univ. Hong Kong, 1994.

[112] R. Tanse, “Distributed genetic algorithms,” in *Proc. 3rd Int. Conf. Genetic Algorithms*, 1989, pp. 434–439.

[113] B. Thomas, *Users Guide for GENEXIS*, System Analysis Research Group, Dept. Comput. Sci., Univ. Dortmund, 1992.

[114] P. W. M. Tsang, “A genetic algorithm for affine invariant object shape recognition,” in *1st IEEE/IEEE Int. Conf. GA's in Engineering Systems: Innovations and Applications*, Sheffield, U.K., 1995, pp. 293–298.

[115] P. Wang and D. P. Kowk, “Optimal design of pid process controllers based on genetic algorithms,” *Contr. Eng. Practice*, vol. 2, no. 4, pp. 641–648, 1994.

[116] M. S. White and S. J. Flockton, “A comparative study of natural algorithms for adaptive IIR filtering,” in *Workshop on Natural Algorithms in Signal Processing*, Chelmsford, Essex, Nov. 1993, pp. 221–228.

[117] D. Whitley, “The GENITOR algorithm and selection pressure: why rank-based allocation of reproductive trials is best,” in *Proc. 3rd Int. Conf. Genetic Algorithms*, J. D. Schaffer, Ed., 1989, pp. 116–121.

[118] ———, “A genetic algorithm tutorial,” Dept. Comput. Sci., Colorado State Univ. Tech. Rep. CS-93-103, Nov. 1993.

[119] B. Widrow, D. E. Rumelhart, and M. A. Lehr, “Neural networks: Applications in industry, business, and science,” *Communicat. ACM*, vol. 37, no. 3, pp. 93–105, Mar. 1994.

[120] D. Wiernek, C. Lucasius, and G. Katenam, “Multicriteria target vector optimization of analytical procedures using a genetic algorithm. Part I. Theory, numerical simulations, and application to atomic emission spectroscopy,” *Analytica Chimica Acta*, vol. 265, no. 2, pp. 211–225, 1992.

[121] P. B. Wilson and M. D. Macleod, “Low implementation cost IIR digital filter design using genetic algorithms,” in *Workshop on Natural Algorithms in Signal Processing*, Chelmsford, Essex, Nov. 1993, pp. 4/1–4/8.

[122] A. H. Wright, “Genetic algorithms for real parameter optimization,” *Foundations of Genetic Algorithms*, J. E. Rawlins, Ed. San Mateo, CA: Morgan Kaufmann, 1991, pp. 205–218.

[123] T. Yamada and R. Nakano, “A genetic algorithm applicable to large-scale job-shop problems,” *Parallel Problem Solving From Nature*, 2, pp. 281–290, 1992.

K. F. Man (M'91), for a photograph and biography, see this issue, p. 518.

Intro.

- 오늘 논문은 **review paper**입니다.
- 대부분의 내용들이 **Appendix와 reference를 바탕으로** 작성되어 있습니다.
(자세하게 더 알아보고 싶은 내용이 있다면 번호로 지정된 해당 논문을 살펴보면 좋을 것 같습니다)
- Genetic Algorithm (GA)에 대한 **전반적인 흐름과 Neural Network와의 비교/분석을 중심**으로 보시면 좋을 것 같습니다.
- 설명 중에 **생물학적인 내용과 단어**들이 자주 사용될 수 있습니다.
(이해가 되지 않는 내용이 있다면 바로 질문해주시면 됩니다!)
- **1990년대 논문**이다 보니 다소 **최신 기술에 대한 적용(Application)이 부족**할 수 있습니다.
(GA의 현재 활용방안에 대해 더 자세히 알고 싶다면 Applications of GA로 검색하시면 됩니다!)

Intro.

Basic Concepts

Theory /
Hypothesis

Structure
Modification

Applications

Conclusions

Basic Concepts

Basic Concepts

Crossover (교차)

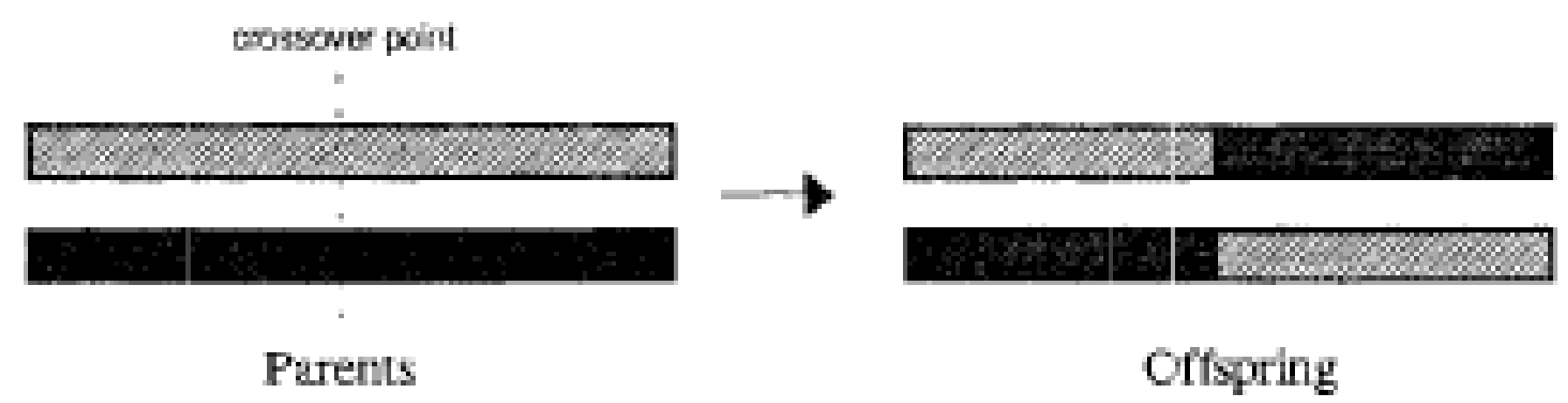
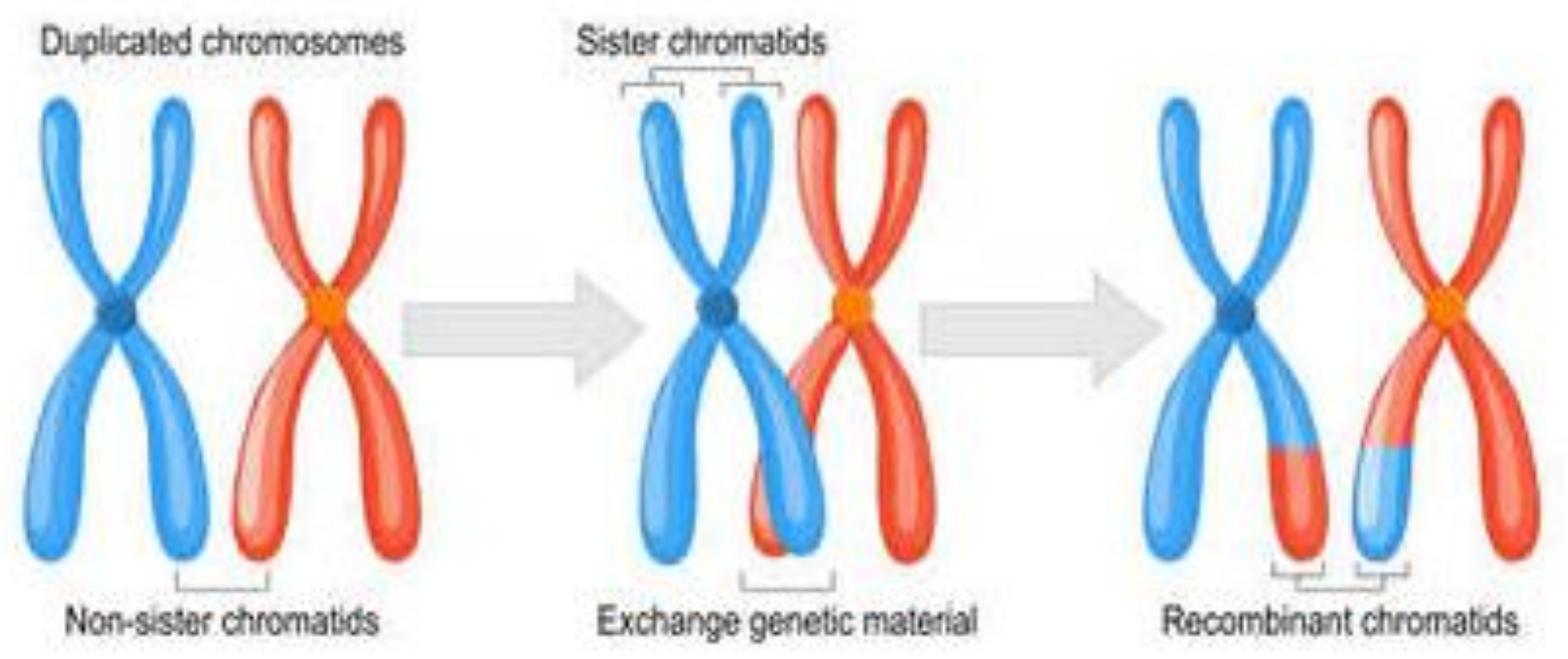


Fig. 1. Example of one-point crossover.

Crossover rate (p_c)

Mutation (돌연변이)

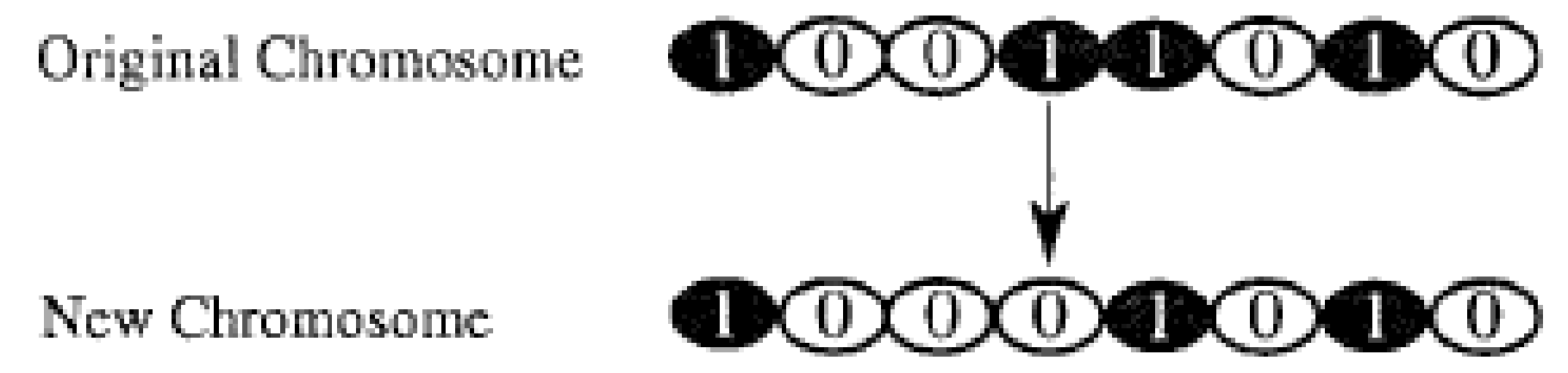
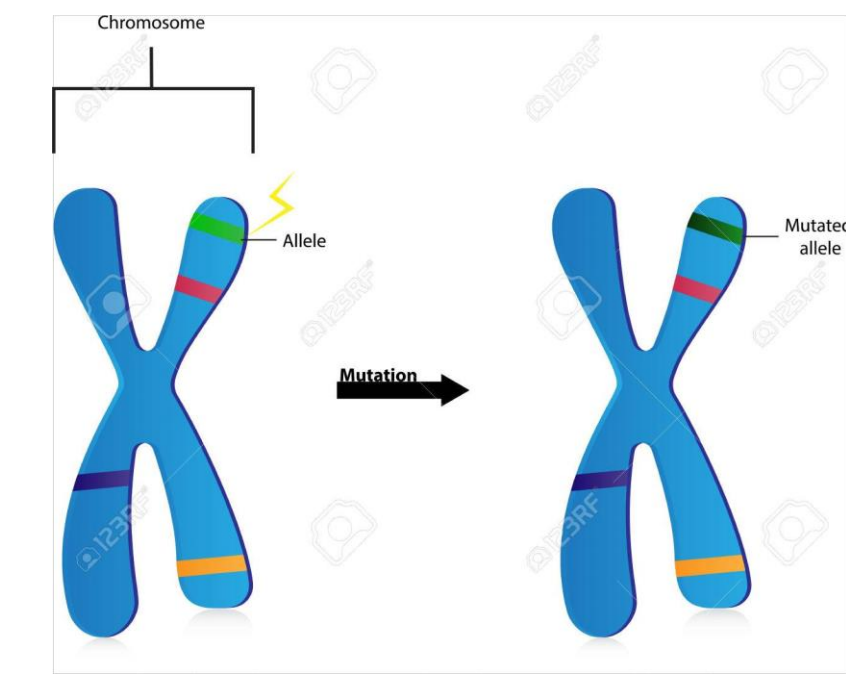


Fig. 2. Bit mutation on the fourth bit.

Mutation rate (p_m)

Basic Concepts

Intro.

Basic Concepts

Theory / Hypothesis

Structure Modification

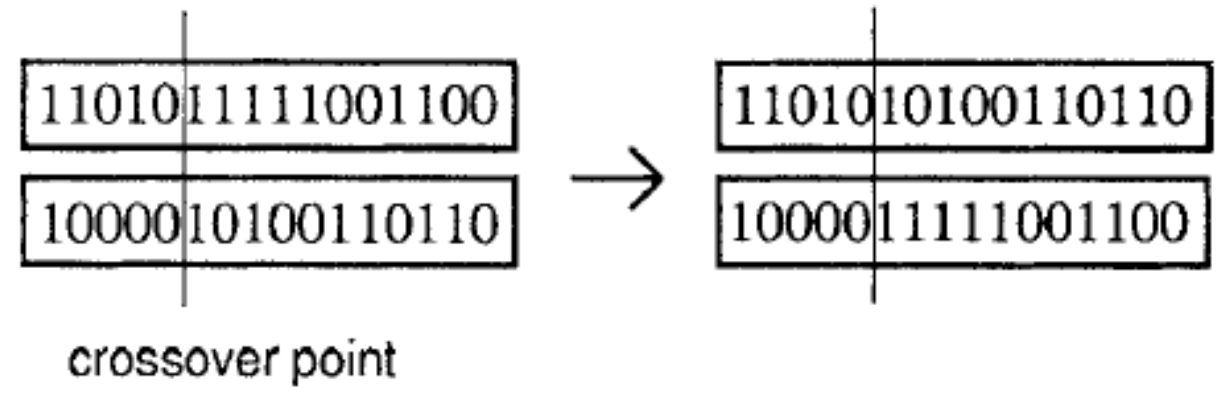
Applications

Conclusions

STEP 1: Parent Selection

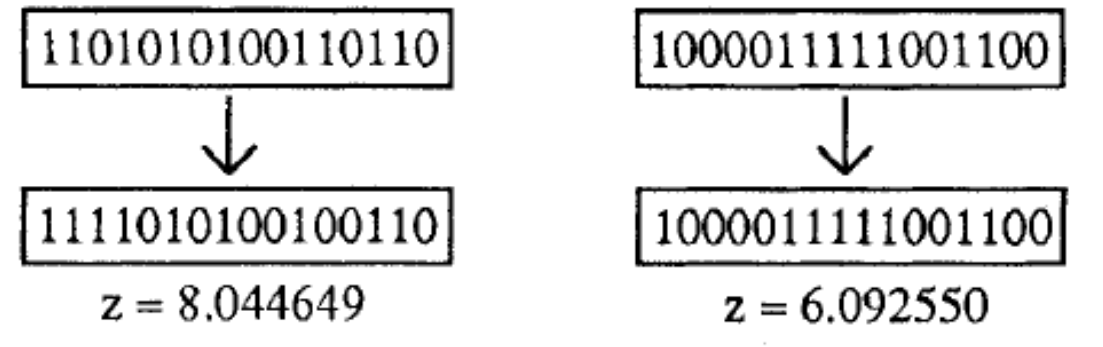
First Population	Objective Value $z = f(x,y)$
1100110110101000	3.481746
0101010110110101	3.668023
1000010100110110	6.261380
1101011111001100	12.864222

STEP 2: CROSSOVER



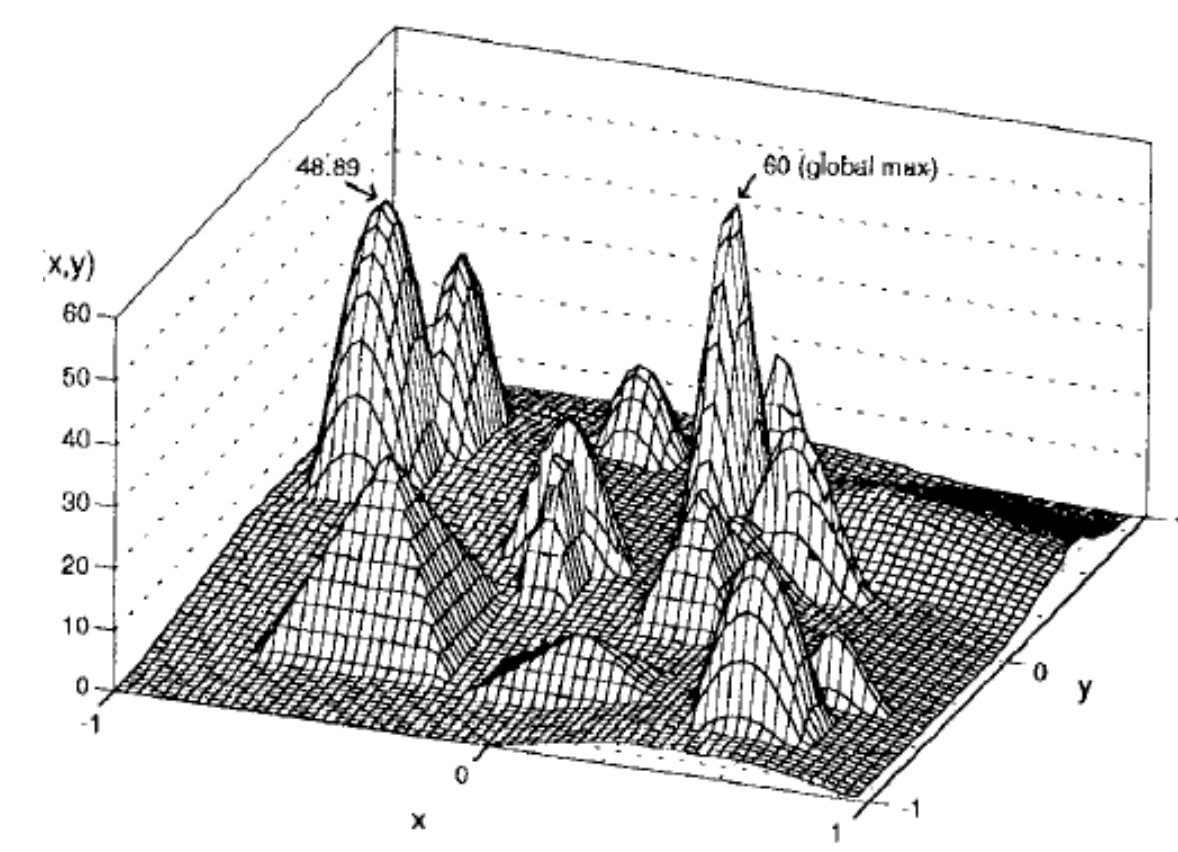
Population size = 4
X(8bit), Y(8bit) → 16bit

STEP 3: MUTATION



STEP 4: Reinsertion

Second Population	Objective Value $z = f(x,y)$
1111010100110110	8.044649
1000011111001100	6.092550
1000010100110110	6.261380
1101011111001100	12.864222



< Question 1 >

Crossover rate (p_c)와 Mutation rate (p_m)의 값의 변화에 따른 후손들의 발생 양상은 어떻게 변화할까?

* case 1, $p_c=1$ 이고 $p_m=0$ 인 경우!

* case 2, $p_c=0$ 이고 $p_m=1$ 인 경우!

**Case 1,
단순 정보의 전달 혹은 재조합**

**Case 2,
무작위 탐색 문제**

Theory & Hypothesis

Theory & Hypothesis

11

Intro.

Basic Concepts

**Theory /
Hypothesis**

Structure
Modification

Applications

Conclusions

Schema Theory

*low-order, above-average **schemata** receive exponentially increasing trials in subsequent generations of a genetic algorithm.*

낮은 차수의 **평균이상의 schemata**는
유전자 알고리즘의 후속세대에서
기하급수적으로 **증가하는 시도를 받는다!**

Schema? sets of strings (encoded form of the chromosome) that have one or more features in common.
형질에 대한 정보를 가지고 있는 문자열

Building Block Hypothesis

*A genetic algorithm seeks near-optimal performance through the juxtaposition of low -order, high-performance schemata, called the **building block***

유전 알고리즘은 **building block**이라고 하는
낮은 차수의 **높은 성능을 가진 schemata**의 **병치**를 통해
최적에 가까운 성능을 찾는다!

Structure Modification

< Question 2 >

GA에서 변경 가능한 부분(조작변인)이 될 수 있는 것은 무엇이 있을까?

13

Intro.

Basic Concepts

**Theory /
Hypothesis**

Structure
Modification

Applications

Conclusions

Structure Modification

1. Chromosome Representation
2. Objective and fitness value
3. Selection mechanism
4. Crossover operations
5. Reordering / inversion
6. Reinsertion
7. Probability rates setting
8. Parallel GA
9. Structed GA

Intro.

Basic Concepts

Theory /
Hypothesis

**Structure
Modification**

Applications

Conclusions

Structure Modification

1. Chromosome Representation

- 보통의 유전자는 **이진데이터로 전환**하여 사용한다.
- 보다 직관적인 **문자열 기반 유전자 표현**도 사용 가능하지만 실제 설계된 GA가 일부 상황에서 **반드시 좋은 결과를 산출하지 못할 수 있다.**
- **다양한 유전자 표현 방식이 존재**하지만 일부 최적화 문제에 대해 어렵고, 때로는 부자연스러운 결과를 초래할 수 있다.

Ex) Order-based representation, Embedded list

First Population
1100110110101000
0101010110110101
1000010100110110
1101011111001100

Intro.

Basic Concepts

Theory /
Hypothesis

**Structure
Modification**

Applications

Conclusions

Structure Modification

2. Objective and fitness value

- 만들어진 유전자(데이터)를 **평가하는 메커니즘은 필수적인 단계**
- 염색체를 입력으로 사용하고 염색체의 성능에 대한 척도로 숫자 또는 객관적인 값 생성
- Linear Scaling
- Power Law Scaling

$$f_i = a \cdot O_i + b$$

$$f_i = O_i^k$$

f_i : fitness value

O_i : Objective value of chromosome i

Structure Modification

4. Crossover operations

- 교차점은 **굳이 한 개일 필요가 없다.** (Multipoint Crossover)
- **Crossover Mask**를 통해 **규칙적**으로 특정지점에 대해서 교차를 진행할 수 있다.
- **교차 방식에 대한 논쟁**도 현재 진행중이다.

Ex) 효과적인 교차 지점의 개수, building block 보존 성능

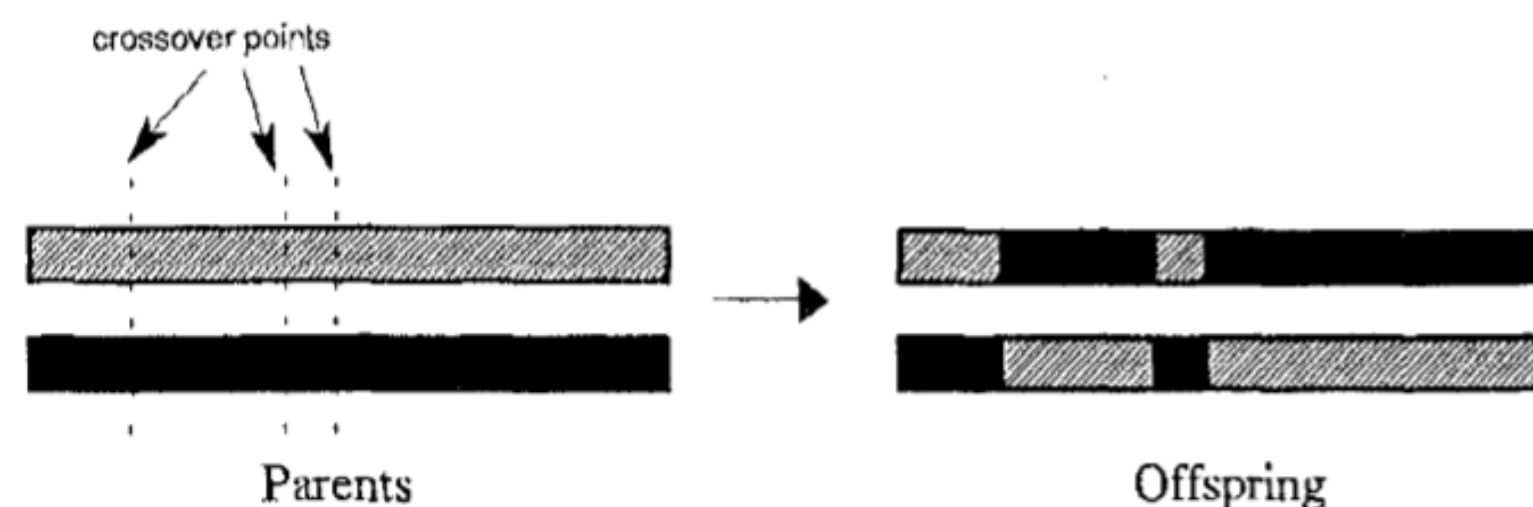


Fig. 7. Example of multipoint crossover, ($m = 3$)

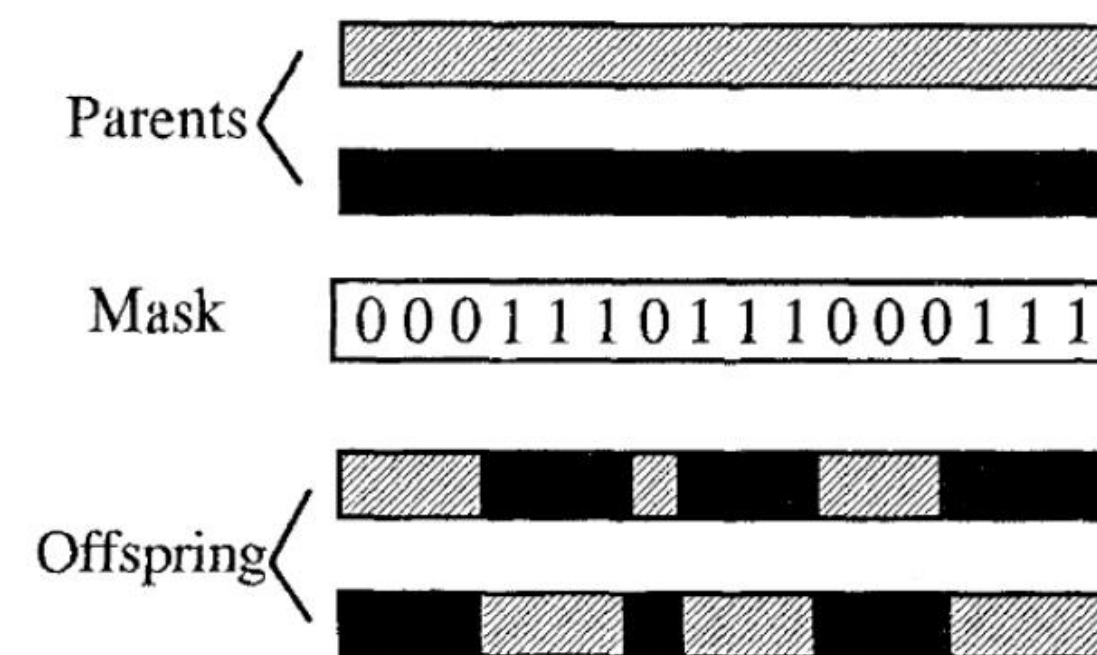


Fig. 8. Example of uniform crossover.

Structure Modification

18

Intro.

Basic Concepts

Theory /
Hypothesis

**Structure
Modification**

Applications

Conclusions

7. Probability rates setting

- GA를 설계할 때 **변경할 수 있는 확률 변수는 p_c 와 p_m** 이다.
- 특정 값으로만 설정하는 것이 정답은 아니기에 상황에 맞게 **적절한 조정이 필요하다.**
- 보편적으로 **$p_m \leq 0.1$** 로 설정하여 과도한 무작위 문제로 바뀌는 것을 방지한다.
- 효과적인 값의 설정에 대한 논쟁도 여전히 진행중이다.

Structure Modification

19

Intro.

Basic Concepts

Theory /
Hypothesis

**Structure
Modification**

Applications

Conclusions

8. Parallel GA

- GA가 받고 있는 비판은 바로 계산에 소요되는 시간이다.
- 하지만 해를 구하는 방정식을 찾는 것보다 수학적이지 않기에 이해가 쉬운 것은 자명하다
- 병렬구조를 활용에 이에 대한 단점을 보완할 수 있다.
- 병렬화 방법에는 전역(Global), 이주(Migration), 확산(diffusion)이 있다.

Structure Modification

8. Parallel GA (Global)

- Global GA는 전체 인구를 **단일 번식 메커니즘**으로 취급한다.
- 공유 메모리 다중 프로세서 또는 분산 메모리 컴퓨터에서 구현 가능하다.
- **Master-Slave 관계를 기반으로 한다.**
(Master가 작업을 처리하는 동안 Slave는 대기해야 한다는 단점이 있다)

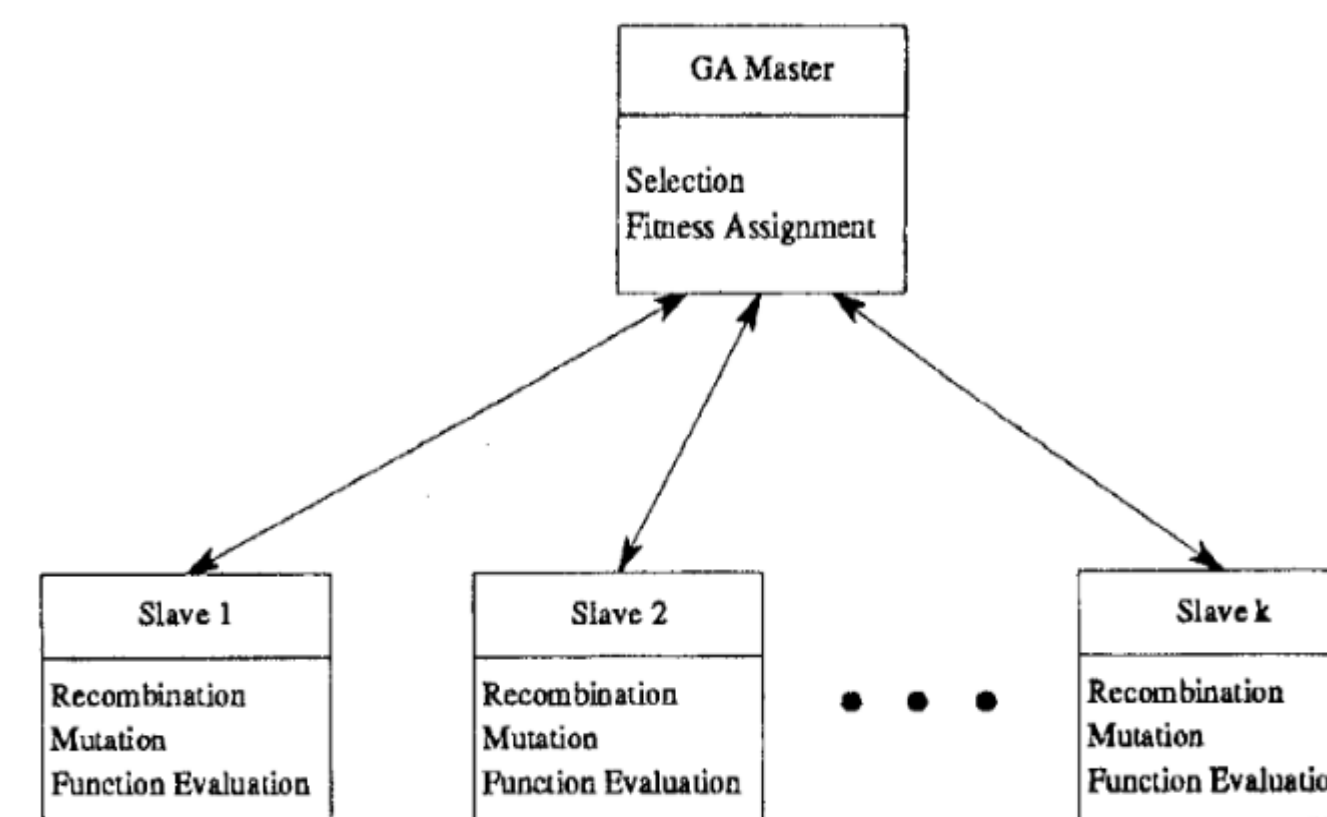


Fig. 9. Global GA.

Structure Modification

8. Parallel GA (Migration)

- Migration GA는 개체군을 여러 하위 개체군으로 나눈다. (각 개체군은 별도의 번식 단위로 처리된다.)
- 하위 개체군 간의 이동은 때때로 발생한다. (이동 방식에 따른 차이가 있다)

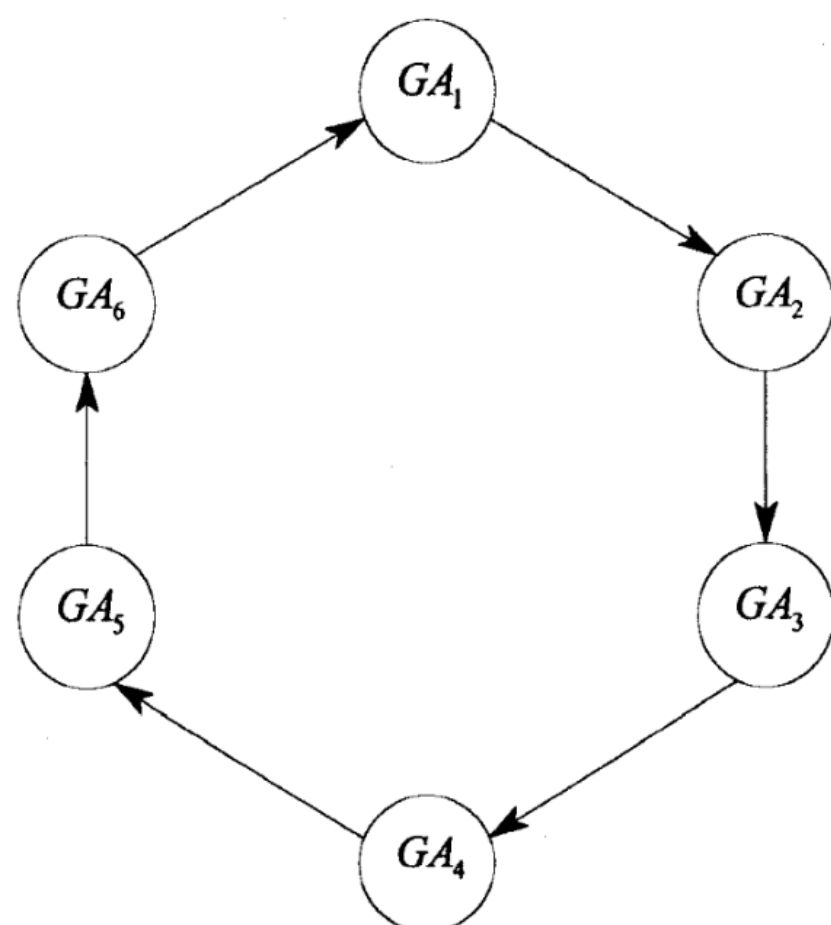


Fig. 10. Ring migration topology.

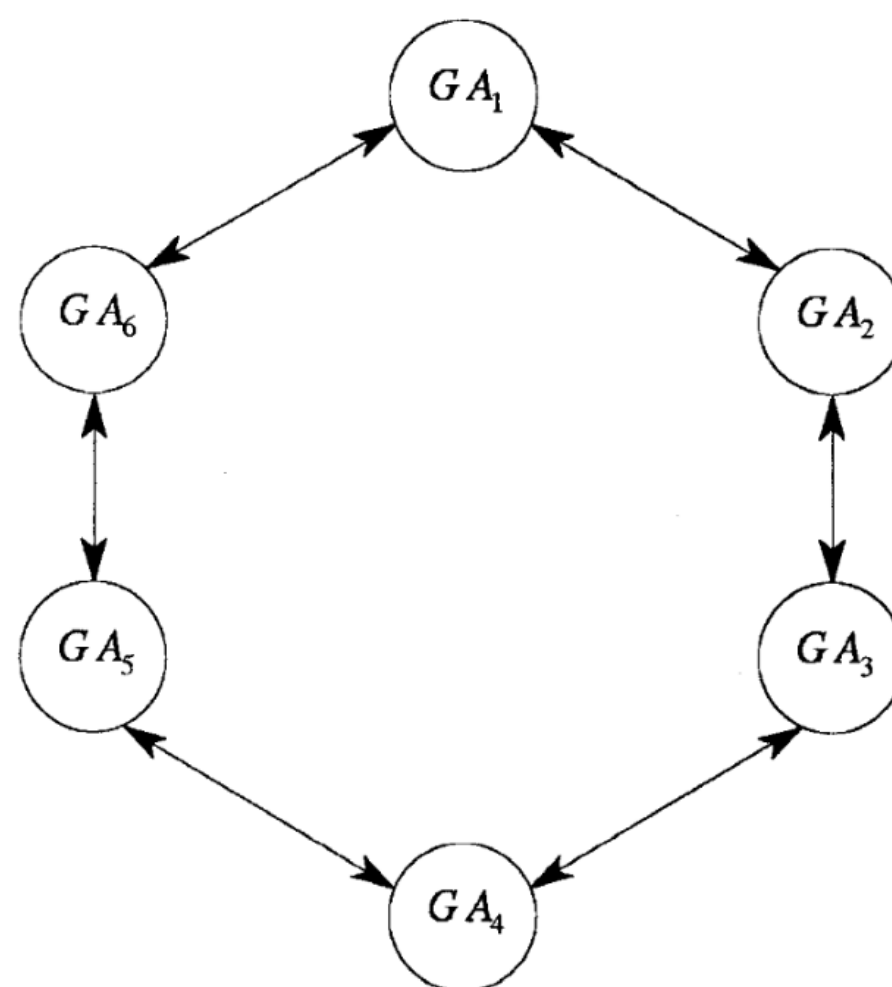


Fig. 11. Neighborhood migration topology.

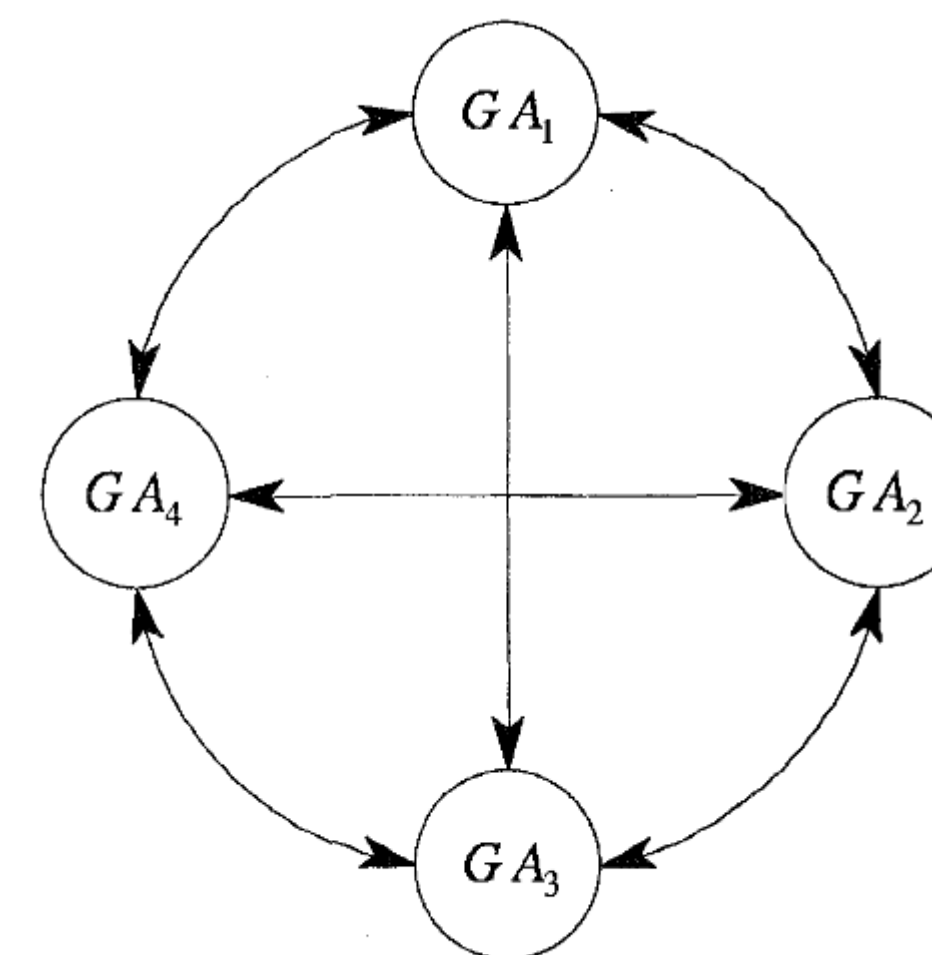


Fig. 12. Unrestricted migration topology.

Structure Modification

8. Parallel GA (Diffusion)

- Migration GA는 인구를 단일 연속구조로 간주한다.
- 2차원 그리드에서 인접한 이웃끼리 데이터의 공유가 가능한 것이 특징이다.

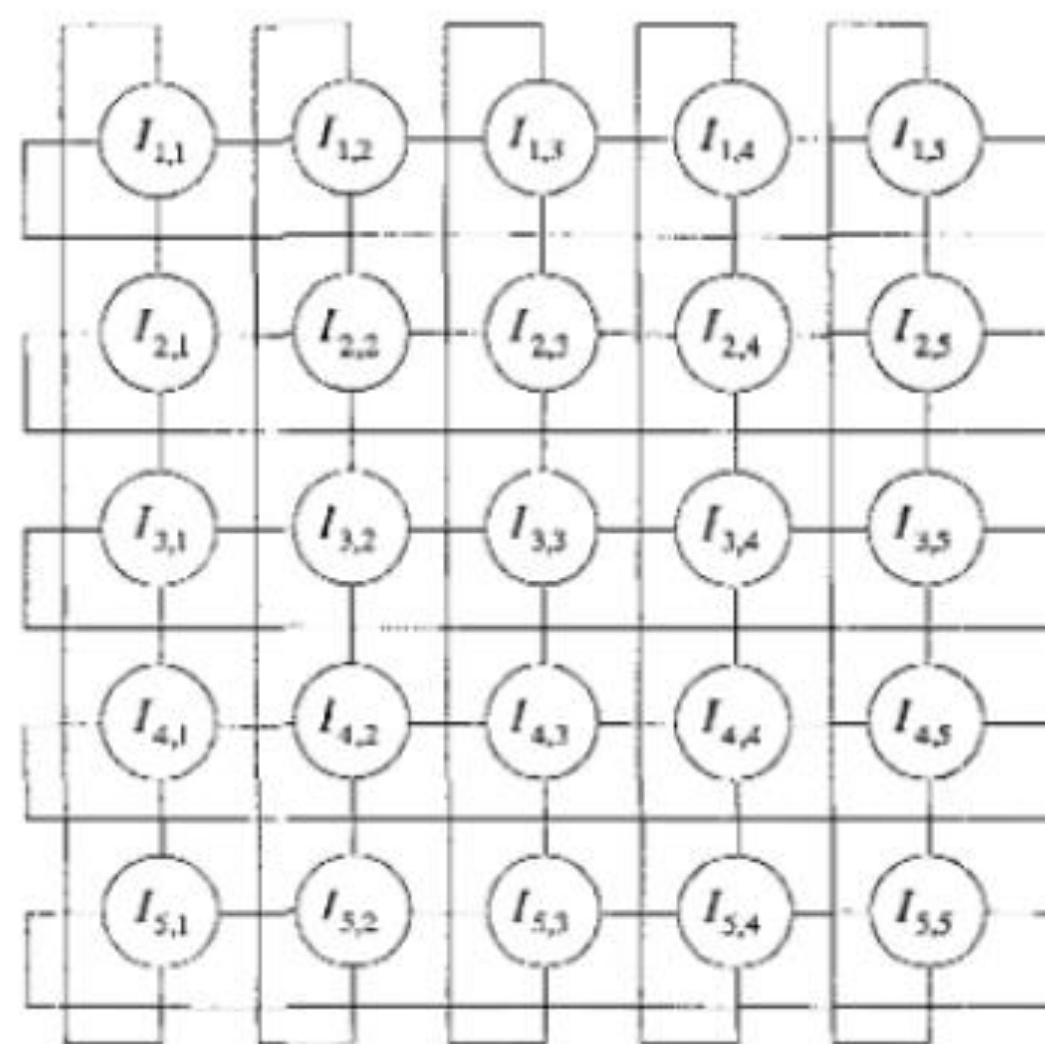
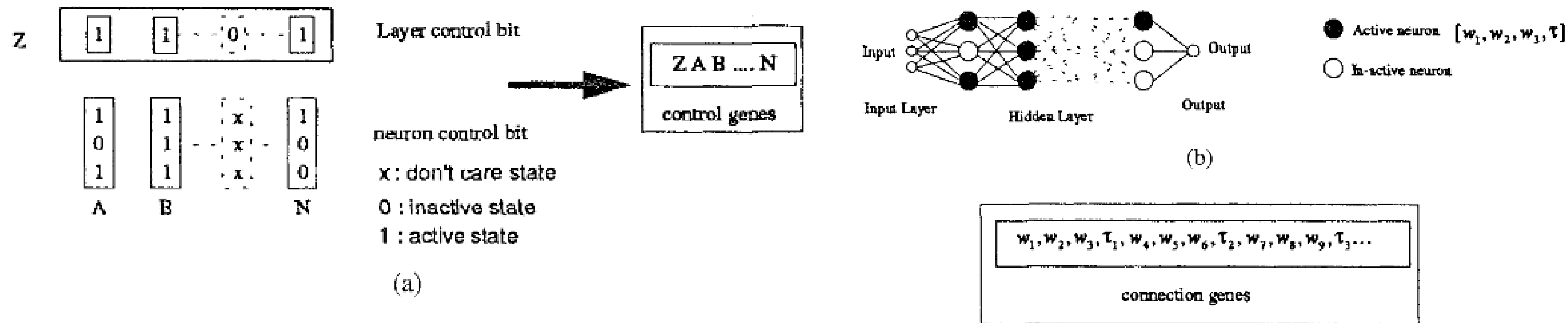


Fig. 13. Diffusion GA.

Structure Modification

9. Structed GA

- Structed GA는 계층구조로 구성된다. (Neural Network와 상당히 유사)
- 구조의 상위 수준 노드는 하위 수준 유전자의 활성화를 제어하게 된다.
(but, 비활성화된 유전자는 변화하는 환경에 반응할 수 있는 추가 정보로 쓰인다.)
- 유전적 다양성을 보존을 위해 하위 구조로 구성된다.



Applications

Applications

1. Parameter and system identification
2. Control
3. Robotics
4. Pattern / Speech recognition
5. Engineering Designs
6. Planning and scheduling
7. Classifier system

25

Intro.

Basic Concepts

Theory /
Hypothesis

Structure
Modification

Applications

Conclusions

Conclusion

Conclusion

- GA 고유의 능력, 특성을 설명하는 시도를 이 논문에서 찾아볼 수 있었다.
- GA에 대한 지식이 거의 없는 사람들에게 이 기술을 소개하는 논문이다.
- GA의 남은 과제는 의심할 여지 없이 실기간 및 적응형 기능이다.

Question?

Thank you