유전 알고리즘을 이용한 스도쿠 퍼즐 생성 및 풀이 방법

[글] 황윤찬
한양대학교 기계공학부
raicing1002@gmail.com

ABSTRACT:

A Sudoku puzzle is a kind of magic square puzzle which requires a non-repeated series of numbers from 1 to 9 in each 9 rows and 9 columns. Furthermore it contains total of 9 small three-by-three matrices, which need non-repeated numbers from 1 to 9 as well. Therefore the total number of possible cases of Sudoku puzzle is finite, even though that of creating nine-by-nine square is exponentially great. Accordingly a certain set of way is need not only for solving the puzzle, but also creating a new one. In this study, the method for creating a Sudoku puzzle applying genetic algorithm is suggested and will be demonstrated. Also, it will be shown that a Sudoku puzzle can be solved by genetic algorithm.

Key Words : Genetic algorithm, Sudoku

1. 서 론

최근에는 시시각각 빠르고 바쁘게 변화하는 일상 속에서 스트레스에 시달리는 사람들이 날이 갈수록 많아지고 있다. 현대 사회에서 살아가는 이상 스트레스를 전혀 받지 않기란 매우 어렵기 때문에, 현대인들은 스트레스의 적절한 관리와 해소를 위해 점점 더 다양한 레저 활동과 취미 생활을 계발하여 정신적 압박에 대처하고 있다. 등산, 조깅 등 신체적 운동을 통해 쾌감을 느끼는 사람들이 있는가 하면 정신적 성취감을 중시하여 보드게임이나 퍼즐과 같은 두뇌활동에 집중하는 부류도 증가하는 추세이다. 최근 오프라인 서점 등에 스트레스 대처를 돕는다는 다양한 퍼즐 관련 상품이 자주 눈에 띄는 것이 이를 보여준다.

매우 유명한 퍼즐인 '스도쿠'는 이런 스트레스 해소용 게임의 좋은 예이다. 처음 주어지는 힌트만 가지고 나머지 빈 칸 모두를 오직 논리와 추리에 의존해 풀어나가야 하는 스도쿠 퍼즐은 기초적인 규칙 외에 다양한 파생 규칙이 적용될 수 있고, 오직 규칙에 따라 퍼즐을 해결하는 일은 항상 만만하지는 않은 작업이다. 또한 일정한 규칙을 따라야 하는 퍼즐이기 때문에 전혀 새로운 스도쿠 퍼즐을 생성하는 것 또한 쉽지 않은 일이다.

본 연구에서는 전통적인 스도쿠 퍼즐의 규칙에 대해 고찰하고 조건에 만족하는 새로운 게임 작성 방법을 고안하기 위하여 MS Visual Studio 6.0을 기반으로 KAIST 플랫폼에서 제공하는 유전 알고리즘인 Simple GA solver를 사용하여 다양한 형태의 스도쿠 퍼즐을 생성하는 방법을 제시한다. 또한 존재하는 실제 문제에 대해 유전 알고리즘을 적용하여 풀이하는 방법 또한 논의한다.

2. 스도쿠 퍼즐

2.1 스도쿠의 정의

스도쿠 퍼즐은 전통적인 마방진의 형태에 스도쿠만의 특별한 규칙을 적용해 만들어진 배열의 임의의 칸을 여러 개 비움으로써 난이도를 조절하며, 처음에 보여지는 숫자들과 해당 게임의 규칙에 따라 빈 칸을 모두 채우게 되면 그 퍼즐을 푼 것이 된다. 일반적으로 가로 9칸, 세로 9칸 총 81칸의 정방형 배열로 구성되어 있으며, 이는 다시 가로 3칸, 세로 3칸 총 9칸의 작은 배열 9개로 다시 구분된다. 스도

5							1		5	7	6	3	9	8	4	1	2
	1			2			3		4	1	8	5	2	7	9	3	6
2			6						2	9	3	6	4	1	5	7	8
	8					3			1	8	2	9	7	6	3	5	4
9					3			1	9	5	7	4	8	3	6	2	1
		4			5		8		6	3	4	2	1	5	7	8	9
					9			5	3	2	1	7	6	9	8	4	5
	4			5			6		7	4	9	8	5	2	1	6	3
	6							7	8	6	5	1	3	4	2	9	7

Fig. 1. An example of Sudoku puzzle and the exact solution, hard difficulty

쿠 퍼즐을 푸는 근본적인 규칙은 일정한 구역에 대해 1부터 9까지 9개의 숫자가 중복되지 않고 나열되어야 한다는 것이다. 전통적인 스도쿠 규칙과 그에 따라 파생될 수 있는 여러 가지 다양한 변형 규칙에 대해서는 2.3절에서 자세히 설명한다.

2.2 스도쿠의 역사

스도쿠의 시초는 중세 아라비아 수학자들이 사용하던 마방진의 한 형태로 알려져 있다. 각 행과 열에 중복되는 문자를 넣지 않고 한 판을 완성하는 이 퍼즐은 수학자 오일러에 의해 18세기에 학문적으로 연구되기 시작하였다. 그러나 이 게임은 마땅한 응용분야가 없어 실용성이 부족하다고 판단되어 주목을 받지 못했다. 이후 20세기 후반에 들어서야 미국과 일본에서 본격적인 퍼즐 게임의 형태로 정립되었으며, 일본 퍼즐 출판사인 '니코리'(Nikoli)가 최초로 '스도쿠'라는 명칭을 부여한 이후로 순식간에 유명세를 타며 전세계로 퍼져나갔다.

오일러가 연구했던 스도쿠 퍼즐의 원형인 '라틴 사각형', 또는 '오일러 사각형'은 다양한 크기의 정방행렬 형태를 취한다. 대체로 사람이 빠르게 형태를 구분하기 쉬운 아라비아 숫자를 사용하는 것이 일반적이지만, 글자나 단어, 문양 등 수학과 관계없는 어떤 것이라도 칸을 채우는 데 사용할 수 있기 때문에 순수 퍼즐 외에 다양한 분야에 응용할 수 있다.

2.3 스도쿠의 규칙

일반적인 스도쿠는 가로와 세로 각각 9칸의 정방형을 취한다. 9개의 행과 9개의 열에는 1부터 9까지 숫자가 중복되지 않고 나열되어야 하고, 여기까지는 전통적인 마방진의 규칙과 비슷하다. 그러나 스도쿠는 특별한 규칙을 더 가지고 있는데, 이 큰 행렬을 다시 9개의 작은 정방행렬로 나누어 각각의 작은 행렬에도 1부터 9까지 숫자가 한 번씩 사용되어야 한다는 것이다. 이 때문에 '숫자는 한 번씩만 쓸 수 있다'라는 뜻의 일본어 문장을 줄인 '스도쿠'가 퍼즐의 이름이 되었다.

새로운 스도쿠 판을 만드는 것은 결코 간단하지 않은 작업이다. 단순히 81개의 칸에 9가지 숫자를 채우는 경우의 수만 고려해도 약 $9^{81} = 1.966 \times 10^{77}$가지 경우가 존재하는데, 이 중에서 스도쿠의 기본 규칙을 만족하는 경우만 추려내는 것은 인력으로는 불가능하며, 컴퓨터를 사용하여 한 경우의 규칙 위반 여부 검사를 1초에 1천만 번 수행하더라도 약 6×10^{68}년이 걸려야 모든 경우를 검사할 수 있다. 따라서 다항함수적 해결이 불가능한 NP문제로 취급되고는 한다. 스도쿠가 오랜 기간 퍼즐로서 가치를 인정받지 못한 것은 이런 비효율성에 기인하며, 컴퓨터까지 동원하여 겨우 실용화가 가능해진 현대에 들어서도 단순히 무작위적으로 판을 생성하는 대신 특별한 알고리즘을 개발하여 규칙적인 방법으로 스도쿠를 생성하는 것이 일반적이다.

이렇게 스도쿠 판이 만들어지고 나면 여기서 임의의 칸들의 숫자를 지워 하나의 퍼즐 게임을 만들게 된다. 여기서 숫자를 얼마나 많이, 또는 어디를 지우느냐에 따라 풀이의 난이도가 결정된다. 항상 어떤 칸을 지우거나 남겨야 한다는 규칙은 없으며, 이 과정은 출제자의 관련 지식과 경험, 직관에 의해 수행된다.

최근에는 비교적 단순한 스도쿠의 전통적 규칙에 지루함을 느낀 퍼즐 애호가들의 노력에 의해 무수히 많은 변형 규칙이 파생되어 구체적인 형태로 정립되어 있다. 중복 불가능 범위를 판의 주대각선까지 포함하는 일명 X-스도쿠 또는 16×16, 25×25등 크기를 확장시킨 빅 스도쿠 등이 가장 간단한 유형의 변형 규칙이다. 더 나아가 주변 칸들과의 대소관계까지 따져야 하는 부등호 스도쿠 같은 최고 난이도의 변형 규칙도 퍼즐 전문지에 심심찮게 소개되고 있다.

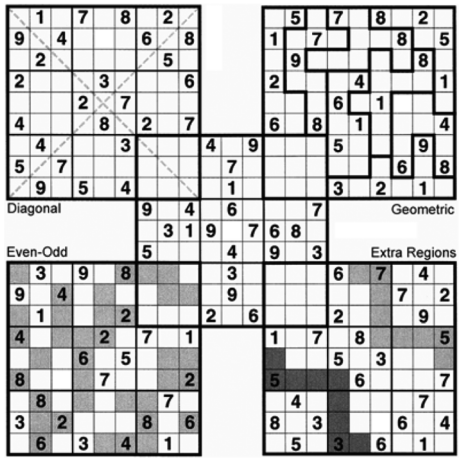

Fig. 2. A hybrid Sudoku puzzle composed of 5 different types, hard difficulty

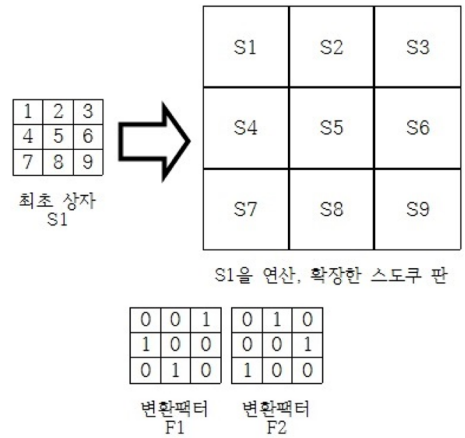

Fig. 3. The demonstration of basic algorithm for creating a Sudoku puzzle

3. 알고리즘

3.1 스도쿠 생성

2.3절에서 상술한 바와 같이 새로운 스도쿠 판을 만들기 위해서는 일련의 알고리즘을 통한다. 가장 많이 사용되는 방법은 특정한 행이나 열, 또는 영역을 무작위적으로 생성한 뒤 그 영역을 적절한 연산과 변형을 통해 9×9 크기의 배열로 확장하는 것이다. 본 연구에서 채택한 방법은 3×3 크기의 작은 배열을 무작위로 생성한 뒤 '변환 팩터'(Transformation Factor)라 하는 2개의 행렬을 이용하여 8개의 변환된 배열을 추가함으로써 크기의 스도쿠 배열을 만든다.

최초로 생성하는 3×3 배열부터 9개의 작은 배열을 각각 S1, S2..... S9라고 한다. 변환에 사용할 두 변환 팩터는 각각 F1과 F2로 둔다. S1은 1부터 9까지 숫자를 무작위적으로 생성하여 만들고 나머지 8개의 배열은 Fig. 3, Table 1와 같이 S1과 변환 팩터를 이용하여 각각의 연산을 거친 뒤 하나의 스도쿠 판으로 재구성된다. 이 때 변환 팩터는 사용자에 따라 다른 값을 사용할 수도 있다.

3.2 유전 알고리즘의 이용

유전 알고리즘은 생물학의 진화 이론을 응용한 연산 방법으로, 임의적으로 생성된 최초의 해집합을 하나의 인구 집단으로 간주하고 적합 선택, 교배, 돌연변이 등 실제 생태계에서 일어나는 현상과 유사한 적응 단계를 거쳐 최종적으로 환경에 가장 잘 적응한, 즉 가장 적합한 해를 구하는 방법이다. 단순히 무작위적인 입력을 제공하는 표준 난수 함수와는 달리 적합도(Fitness)라는 개념을 적용하여 계산이 거듭될수록 효율적으로 유용한 해에 접근할 수 있다는 것이 가장 큰 장점이다. 이를 이용하여 스도쿠의 기본 규칙 뿐만 아니라 원하는 변형 규칙을 따르는 고급 스도쿠도 생성할 수 있다.

난수를 발생시키는 특성을 가지고 있으므로 스도쿠 퍼즐에 있어서는 생성뿐만 아니라 실제 빈칸과 초기 힌트가 주어진 하나의 문제를 해결하는 데에도 유용한 도구가 될 수 있다. 특히 적합도의

판단이라는 강력한 특성을 이용하면 손으로 풀기 매우 어려운 난이도의 퍼즐이라도 유전 알고리즘을 이용하여 빠르게 풀 수 있다.

4. 유전 알고리즘을 적용한 스도쿠 생성 및 풀이 방법의 구체적 구현

Table 1. Constructing 8 other boxes from the first box S1 by matrix multiplication with transformation factors

S2	F2 * S1
S3	F1 * S1
S4	S1 * F1
S5	F2 * S1 * F1 = S2 * F1
S6	F1 * S1 * F1 = S3 * F1
S7	S1 * F2
S8	F2 * S1 * F2 = S2 * F2
S9	F1 * S1 * F2 = S3 * F2

4.1 스도쿠의 프로그래밍

스도쿠는 수학적 원리가 아닌 논리적 규칙에 의해 만들어지고 풀이된다. 따라서 C++언어를 통해 스도쿠의 기본 규칙을 구현하려면 생성된 판의 각 행, 열과 3 × 3 배열(이하 '상자')에 숫자 중복이 발생하는지 판단하는 과정이 필요하다. 스도쿠 판의 N개의 빈칸을 구하고자 하는 N개의 해로 두고 판을 생성한 뒤 모든 행, 열과 상자의 숫자 중복 여부를 확인하여 중복이 적을수록 적합도가 높아지도록 구성하였다.

본 연구에서는 스도쿠 생성 및 풀이에 대한 유전알고리즘의 적용 실례를 보이는 것을 목적으로 하므로, 변형 규칙을 제외한 스도쿠 기본 규칙을 만족하는 퍼즐을 구성하도록 하였다.

GA solver의 입력은 염색체의 수는 population size = 500, 세대수는 generation number = 500으로 두었으며 교배율은 0.8, 돌연변이율은 0.005로 지정했다.

4.2 스도쿠 생성

본 연구에서 사용하도록 지정된 Simple GA solver로는 해공간을 이산적으로(discrete) 탐색하는 것이 불가능하다. 따라서 동작 과정에서는 부동소수점이 포함된 해집합을 바닥 함수를 사용해 강제 이산화하는 방식을 사용하였다.

3.1절에 소개한 변환 팩터 방법을 이용하기 위해 최초의 상자 S1을 유전 알고리즘으로 생성, 연산을 통해 확장하여 스도쿠 판을 만든다. 이 때 최종적으로 생성된 판에 중복 검사를 시행하여 적합도를 판단하게 되는데 최초의 해집합은 S1의 각 칸에 숫자가 무작위로 부여되므로 이 상태에서의 적합도 검사는 단순히 중복 여부 검사와 크게 다르지 않은 수준에 불과하다. 그러므로 적합성 판단의 활용을 위해 1행의 각 열에 1부터 9까지 순서대로 들어간 판이 최적해로 유도되도록 하여 결과를 도출하였다. 이렇게 만들어진 스도쿠 판은 Fig.1와 같고, 적합도 판단 과정은 Table.2에 나타내었다.

1	4	7	2	5	8	3	6	9
2	5	8	3	6	9	1	4	7
3	6	9	1	4	1	2	5	8
4	7	1	5	8	2	6	9	3
5	8	2	6	9	3	4	7	1
6	9	3	4	7	1	5	8	2
7	1	4	8	2	5	9	3	6
8	2	5	9	3	6	7	1	4
9	3	6	7	1	4	8	2	5

Fig. 4. Full Sudoku puzzle created by proposed algorithm and simple GA solver

Table 2. Performance measure of proposed algorithm by simple GA solver

Generation	Fitness (Max)	Fitness (Min)	Fitness (mean)	Deviation
0	632	361	578.01	33.47
50	655	487	614.04	18.13
100	657	579	635.76	15.34
150	657	581	642.64	12.30
200	657	598	644.45	11.23
250	657	597	645.21	11.30
300	657	616	645.85	10.00
350	657	598	645.76	10.50
400	657	615	645.57	9.64
450	657	599	645.72	10.63
500	657	598	645.81	10.20

*Theoretical Max. Fitness = 657

3	9	6	4	7	1	2	5	8
7	5	2	9	3	8	6	4	1
8	1	4	5	6	2	3	9	7
	3	9	1	2		7	8	6
2				4	3	9		5
1	7						2	3
9	8						7	
5				1	9		6	4
	4	1	2	8				9

Fig. 5. A Sudoku puzzle with 25 blanks

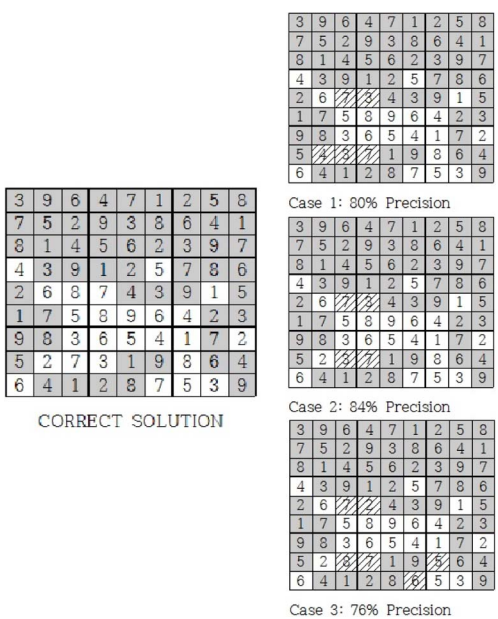

Fig. 6. The correct solution of Fig.5 and several outputs from the proposed algorithm

4.3 스도쿠 풀이

초기 힌트와 빈칸이 주어져 있는 실제 스도쿠 문제를 푸는 데에도 유전 알고리즘이 응용될 수 있다. 다만 상술한 바와 같이 현재 플랫폼에서는 해집합의 이산적 탐색이 불가능하므로 의미 없는

해공간, 즉 자연수가 아닌 수에 대해서도 GA solver가 해집합을 탐색하게 되어 불필요한 계산이 많이 시행된다. N개의 빈칸을 가진 스도쿠를 풀기 위해서 N개의 해를 지정하고 탐색하는데 N이 커질수록 해집합의 크기가 증가하여 결과의 완성도가 낮아지게 된다. 예시로 보일 문제는 Fig. 4과 같이 25개의 빈칸을 가진 실제 문제이다. 해당 문제의 실제 정답 및 본 연구에서 설계한 방법으로 풀이한 결과의 비교를 Fig. 6에 나타내었다. 음영으로 구분된 칸은 초기 힌트이며, 음영이 없는 25개의 칸이 탐색하는 해집합이다. 좌측의 정답과 비교하여 오답이 도출된 칸은 빗금으로 표시하였다. Fig. 6에 나타난 결과의 정확도는 정확한 값을 얻은 칸의 개수에 따라 나타내었다. Table 3에는 스도쿠 풀이 작업의 적합도 판단 과정이 나타나 있다.

5. 결 론

본 연구에서는 스도쿠 퍼즐의 기본 규칙에 부합하는 새로운 게임 생성 및 실제 문제 풀이 방법에 유전 알고리즘을 적용하는 방법에 대해 논의하였다. C++ 프로그램을 통하여 작성된 실례와 문제 풀이에의 적용 결과를 제시하였고 다음과 같은 결론을 도출하였다.

(1) 스도쿠 생성 기법으로서 제시된 알고리즘으로 작성된 스도쿠는 그 거시적인 형태에 있어서 언뜻 구별 가능한 패턴이 관찰될 수 있으며, 결과의 다양성은 3×3 크기의 최초 상자 S1에만 의존하기 때문에 무작위적 난수 생성에 비해 비교적 단순해 보일 수 있다. 그러나 상자 하나에 9가지의 숫자를 나열하는 경우의 수만 세어도 9! = 362,880가지에 달하므로 실용적인 퍼즐 생성 방법이라는 목적을 고려하면 결코 성능이 부족하지 않다고 판단한다. 또한 유전 알고리즘의 적합성 판단 능력이라는 특성을 적용하면 기본 규칙 외에도 특정한 조건을 만족하는 게임도 생성할 수 있다. 따라서 새로운 스도쿠 게임의 생성에 있어서 유전 알고리즘의 활용 가능성을 보였다.

(2) 스도쿠 문제 풀이 기법으로서 유전 알고리즘을 통해 도출된 해답은 실제 정답과 완벽히 일치하지는 않는다. 프로그램 동작 중 정확한 정답이 출력되는 경우도 있으나 수많은 Case 중의 일부로서 존재할

Table 3. Performance measure of proposed solution method by simple GA solver

Generation	Fitness (Max)	Fitness (Min)	Fitness (mean)	Deviation
0	936	892	918.49	6.76
50	972	947	962.07	3.36
100	972	957	968.00	2.93
150	972	955	967.55	3.31
200	972	957	967.95	3.29
250	972	958	968.28	2.93
300	972	959	968.02	2.87
350	972	957	968.42	2.92
400	972	956	967.76	3.16
450	972	956	968.42	2.90
500	972	958	967.90	3.08

*Theoretical Max. Fitness = 972

뿐, 유전 알고리즘의 목적인 '최적해로의 수렴'이라는 측면에서는 의미가 부족한 결과라고 판단하였다. 따라서 존재하는 스도쿠 문제의 풀이에 유전 알고리즘을 적용하는 것이 이론적으로 가능하나, 완벽한 정답을 만족하는 해를 구하기는 어려울 수 있음을 관찰하였다. 본 연구를 위해 제공된 Simple GA solver는 구조적으로 해집합을 연속적인 해공간에서만 탐색하도록 구축된 상태이므로 이산적인 자료 형태를 요하는 스도쿠 문제에 적용하기에는 다소 적합한 형태가 아니라고 해석할 수 있다. 그러나 4.3절의 Fig. 6을 보면 빈칸을 정확히 맞힌 비율이 상당히 높은 편인데, 이는 사용환경을 개선한다면 유전 알고리즘을 충분히 실용적으로 스도쿠 풀이에 사용할 수 있다는 가능성을 제시한다.

유전 알고리즘의 적합성 판단 기능을 이용하여 간단한 퍼즐의 생성이 가능하고 유용한 해답을 도출할 수 있는 능력이 있음을 확인하였다. 해집합을 이산적인 해공간에서 탐색할 수 있도록 개량하는 등 본 연구에서 제시한 방법의 컴퓨터 프로그램 구현 완성도를 개선한다면 계산 능력을 보다 효율적으로 활용하여 더욱 유의한 수준의 결과를 얻을 수 있을 것으로 기대한다.

감사의 글

본 연구는 미래창조과학부가 주최하는 제4회 EDISON SW 경진대회 전산설계분야 참가를 목적으로 계획수립, 진행하였다. 개인적으로 컴퓨터 프로그래밍은 항상 어렵고 난해한 분야이기 때문에 프로젝트를 시작하면서 주제 선정부터 매우 난감하고 힘겨웠으며, 코딩 작업하는 내내 여러 번 막다른 길을 맞아 수 차례 연구 기획부터 다시 시작하는 등 벅찬 위기가 많았다. 그럼에도 불구하고 벌써 세 번에 걸쳐 괄목할 만한 성과를 낸 EDISON 경진대회에 참가할 기회를 주시고 많은 지도와 도움을 주신 한양대학교 Multiphysics Systems Lab 윤길호 교수님, 최정식 조교님께 감사 드린다. 또 유전 알고리즘이라는 생소한 시스템을 제대로 활용할 수 있도록 조언해주신 KAIST MDPS Lab 여승균 연구원님께도 감사 드리고 싶다.

스도쿠 등 두뇌퍼즐은 평소에도 취미로서 관심이 매우 많은 분야였는데, 이번 프로젝트를 진행하면서 스도쿠를 만들고 풀 수 있는 다양한 기법과 노하우를 접할 수 있어 매우 유익한 기회가 되었다. 다만 스도쿠 문제 풀이가 처음 생각만큼 완벽하지 않았던 점은 못내 아쉬우며, 기회가 된다면 더 개선된 플랫폼에서 다시 이 방법을 시험해볼 수 있기를 바란다.

참고문헌

1. 문병로, 2008, 쉽게 배우는 유전 알고리즘: 진화적 접근법, 한빛미디어, pp. 19-33, 169-177.
2. 김여근, 2011, 진화 알고리즘, 전남대학교출판부, pp. 59-74.
3. 최윤경, 2004, 유전 알고리즘을 이용한 최적화 방법에 관한 연구, 한양대학교 백남학술정보관.
4. 스도쿠퍼즐 생성 알고리즘, http://www.cyworld.com/vbman/3359600.