DOI: 10.3745/KIPSTB.2008.15-B.2.131

유전자 알고리즘을 이용한 그래프에서 $L(2,1)$-labeling 문제 연구

한 근 희[†] · 김 찬 수[††]

요 약

그래프 G = (V, E) 의 $L(2,1)$-labeling 이란 함수 f: $V(G) \rightarrow \{0, 1, 2, ...\}$ 를 정의하는 것으로서 함수 f 는 만일 G 내의 두 개 정점 u, v 사이의 최단거리가 1 인 경우 $|f(u) - f(v)| \geq 2$ 라는 조건 및 최단거리가 2 인 경우 $|f(u) - f(v)| \geq 1$ 라는 조건을 만족시켜야 한다. $\lambda(G)$ 로 표기되는 G 의 $L(2,1)$-labeling 수는 모든 가능한 f 들 사이에서 사용된 가장 큰 정수가 가장 작은 값을 나타낸다. 상기한 문제는 NP-complete 계열의 문제이기 때문에 본 논문에서는 $L(2,1)$-labeling 에 적용 가능한 유전자 알고리즘을 개발한 후 개발된 알고리즘을 최적값이 알려진 그래프들에 적용하여 그 효율성을 보이고자 한다.

키워드 : $L(2,1)$-labeling, 유전자 알고리즘, 그래프

Solving $L(2,1)$-labeling Problem of Graphs using Genetic Algorithms

Han, Keunhee[†] · Kim, Chansoo[††]

ABSTRACT

$L(2,1)$-labeling of a graph G is a function f: $V(G) \rightarrow \{0, 1, 2, ...\}$ such that $|f(u) - f(v)| \geq 2$ when $d(u, v) = 1$ and $|f(u) - f(v)| \geq 1$ when $d(u, v) = 2$. $L(2,1)$-labeling number of G, denoted by $\lambda(G)$, is the smallest number m such that G has an $L(2,1)$-labeling with no label greater than m. Since this problem has been proved to be NP-complete, in this article, we develop genetic algorithms for $L(2,1)$-labeling problem and show that the suggested genetic algorithm peforms very efficiently by applying the algorithms to the class of graphs with known optimum values.

Key Words : $L(2,1)$-labeling, Genetic Algorithms, Graphs

1. Introduction

Let $G = (V, E)$ be a simple graph. $L(2,1)$-labeling of G is an function f: $V(G) \rightarrow [0, \infty)$ satisfying the following two conditions:

(i) if $(x, y) \in E$, then $|f(x) - f(y)| \geq 2$, where $x, y \in V$,
(ii) if $d(x, y) = 1$, then $|f(x) - f(y)| \geq 1$, where $d(x, y)$ denotes the shortest distance between two vertices x and y.

The $L(2,1)$-labeling number of G is the smallest number m such that G has an $L(2,1)$-labeling with no label greater than m and is denoted by $\lambda(G)$ of G.

〈Table 1〉 [4] Upper bounds of $\lambda(G)$ for various class of graphs and their complexities

Graphs	Upper bounds	Complexity
Strongly Chordal	$\lambda(G) \leq 2\Delta$	
Permutation	$\lambda(G) \leq 5\Delta - 2$	
Planar	$\lambda(G) \leq 2\Delta + 25$	NP-complete
Split	$\lambda(G) = \Theta(\Delta^{1.5})$	NP-complete
Chordal	$\lambda(G) \leq (1/4)(\Delta + 3)^2$	NP-complete
Bipartite	$\lambda(G) = \Theta(\Delta^2)$	NP-complete
Graphs of diameter 2	$\lambda(G) \leq \Delta^2$	NP-complete
General graphs	$\lambda(G) \leq \Delta^2 + \Delta$	NP-complete

† 종신회원 : 공주대학교 응용수학과 부교수
†† 정 회 원 : 공주대학교 응용수학과 조교수
논문접수 : 2007년 10월 30일, 심사완료 : 2008년 1월 3일

$L(2,1)$-labeling of graph has its applications in the area of radio frequency assignment problems. Roberts [2], in order to avoid inteference between transmitters, proposed the problem of efficiently assigning radio channels to transmitters at several locations, using nonnegative integers to represent channels, so that closely located transmitters receive different channels, and very closely located transmitters receive channels at least two apart. Later, this problem was formulated as $L(2,1)$-labeling problem of graphs by Griggs et. al. [6]. Note that $L(2,1)$-labeling is a special case of so called $L(j, k)$-labeling of graphs [3]. $L(j, k)$-labeling is defined as a function: f: $V(G) \rightarrow \{0, 1, 2, ...\}$ such that $|f(u) - f(v)| \geq j$ when $d(u, v) = 1$ and $|f(u) - f(v)| \geq k$ when $d(u, v) = 2$.

However, determining $\lambda(G)$ for general graphs has been proved to be NP-complete by Griggs et. al. [6]. This problem also remains NP-complete for more restricted class of graphs. Therefore, researches have focused on determining the upper bound of $\lambda(G)$ using Δ as the parameter, where Δ is the maximum degree of a graph G. Bodlaender [4] summarized the upper bounds of $\lambda(G)$ and status of compexities of various class of graphs as in <Table 1>. We note that the upper bound of general graphs has been reduced to $\Delta^2+\Delta-2$ by Goncalves [1].

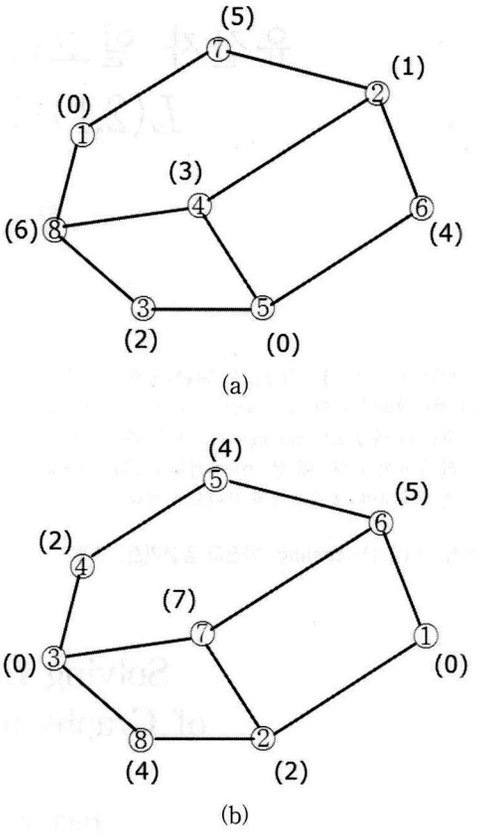
Genetic algorithms (GAs), a part of evolutionary computing, were introduced by Holland in 1975 [5]. Since then genetic algorithms have been applied to a large variety of combinatorial optimization problems.

In this paper, our main objective is to develop Genetic Algorithms (GAs) which can be applied to $L(2,1)$-labeling problems. In section 2, we discuss the properties of $L(2,1)$-labeling of graphs, and describe our stratigies of genetic algorithms for $L(2,1)$-labeling problems. In section 3, we apply the suggested algorithms to several complete k-partite graphs and analyze the performance of the algorithms. Section 4 contains the conclusions.

2. Genetic algorithms for L(2,1)-labeling Problems

```
Algorithm First-fit (FF)
Input: A graph G(α) with an ordering α = (v1,
       v2, ..., vn) of vertices.
S = {x| x is an integer ≥ 0}.
Output: L(2,1)-labeling of G
begin
  for i = 1 to n do
    label vi by the lowest possible integers in S;
end
```

(Fig. 1) First-fit Algorithm



(Fig. 2) Applications of FF algorithm on a graph with different orderings. The numbers inside the parenthesis are the corresponding $L(2,1)$-labelings. The largest labels assigned in (a) and (b) are 6 and 7, respectively

For a graph $G = (V, E)$, an ordering of V is a bijection $\alpha:\{1, 2, ..., n\} \leftrightarrow V$, where $n = |V|$. We let $G(\alpha)$ be an ordered graph by some ordering α of V. $L(2,1)$-labeling of G can be computed as follows: first we order the vertices of G by some ordering α. Next, label the vertices in succession by the lowest possible integer. These procedures are known as *First_fit* (*FF*) algorithm and formally presented in (Fig 1). (Fig. 2) shows the applications of *FF* algorithm on a graph.

(Fig. 2) shows that application of *FF* algorithm on a graph G with different orderings yield different $L(2,1)$-labelings. Let $max(G, \alpha)$ be the maximum label obtained from the graph $G(\alpha)$ after applying algorithm *FF*. Since different orderings yield different $max(G, \alpha)$ for a given graph G, our genetic algorithm searches for the ordering that produces minimum $max(G, \alpha)$.

2.1. Representations of chromosomes

Since we are searching for the optimum ordering of the vertices, for encoding, we use the *path representation* originally developed for the TSP (Travelling Salesman Problem)[5]. Path representation is permutation of the

integers [1..n], where n is the number of vertices of G. For example, if n = 6 then the chromosome (3, 2, 1, 6, 4, 5) represents the ordering α of the vertices. Note that even though path representation is the most natural representation of the orderings of vertices, it does not allow classical crossovers and mutations.

2.2. Selection

Let *pop* represents the population of chromosomes and $L_v(c, G)$ represent $L(2,1)$-labeling number of a vertex v as a result of applying algorithm *FF* to a chromosome c. Also, let $L_{max}(c, G) = \max\{L_v(c, G) | v \varepsilon V\}$. Then for each $c \varepsilon pop$, the fitness function $val(c)$ is defined as follows:

$$val(c) = \max\{L_{max}(c, G) | c \varepsilon pop\} - L_{max}(c, G).$$

Note that even though the $L(2,1)$-labeling problem is a minimization problem, our fitness function $val(c)$ allows to treat it as a maximization problem. For the selection we use *roulette wheel* selection mechanism with slots sized proportionally to the fitness of chromosomes. Our genetic algorithm also enforces *elitism* which is a mechanism to ensure that the most highly fit chromosomes of the population are passed on to the next generation.

2.3 Genetic operators

For the crossover, we use the well known three crossover operators; *partially mapped* (PMX), *order* (OX) and *cycle* (CX) crossover. We also adapt six mutation operators; *displacement*(DM), *exchange* (EM), *inversion* (IVM), *scramble* (SM), *insertion* (ISM) and *simple-inversion* (SIM) mutation. Among these operators *cycle* crossover and *simple-inversion* mutation are the two most important operators in this paper. Hence we include the mechanisms of these operators for the completeness of this paper. The mechanisms of other operators can be found in [9, 8].

The CX crossover operator [7] builds offspring in such a way that each element of offspring inherits from one of the parents. Let p_1 = (1 2 3 4 5 6) and p_2 = (5 6 2 1 4 3) be the two parents and o_1 be the offspring. The first element of o_1 is the same as the first element of p_1. Hence, o_1 becomes (1 x x x x x), where x means "not known". The element of p_2 below the 1 in p_1 is 5 and 5 is positioned as the fifth element of p_1. Hence, the fifth element of o_1 becomes 5 which yields o_1 = (1 x x x 5 x). This, in turn, implies the element 4 must be the fourth element of o_1 which yields o_1 = (1 x x 4 5 x). Continuing this rule, element 5 must be included as the fifth elemnet of o_1. However, 5 is already in the o_1. Thus the remaining elements are filled from the p_2. Therefore, o_1 becomes (1 6 2 4 5 3).

The SIM mutation operator [5] requires two cut points in the chromosome and reverse the substring between these two cut points. For example, consider chromosome (1 2 3 4 5 6), and suppose that the first cut point is chosen between first and second element, and the second cut point is between fourth and fifth element. Then the result of SIM operation is chromosome (1 4 3 2 5 6).

3. Experimental Results

A *k-partite graph* is a graph whose vertices can be partitioned into k disjoint sets so that no two vertices whthin the same set are adjacent. A *complete k-partite graph* is a k-partite graph such that every pair of vertices in the k sets are adjacent. If there are p, q, ..., r vertices in the k sets, the complete k-partite graph is denote $K_{p,q,...,r}$. Note that bipartite graph is a special case of k-partite graph with k = 2.

As far as we know there has been no known published results for applying genetic algorithms to $L(2,1)$-labeling problems. Therefore, in order to measure the efficiency of our genetic algorithms, we apply our genetic algorithm to graphs with known $\lambda(G)$ values. Complete k-partite graphs are such graphs and their $\lambda(G)$ values, first proved by Griggs [6], are shown in Theorem 1.

We generated three complete k-partite graphs with different partite sets. <Table 2> shows these three test graphs with their various graph properties.

The $\lambda(G)$ of complete k-partite graphs can be obtained as follows. Let G be a complete k-partite graph. Since the shortest distance between any two vertices in G is at most 2, for the $L(2,1)$-labeling for G, all the labels must be distinct. Furthermore, consecutive labels can not be assigned at vertices from different partite sets. Therefore, it is esay to verify that the following algorithm shown in

〈Table 2〉 Three test graphs and their properties. n, m are the number of vertices and edges, respectively while k is the number of partite sets

Graphs	n	m	k	$\lambda(G)$
partite1 = $K_{6,6,6,6,6}$	30	360	5	33
partite2 = $K_{10,\ 10,\ 10,\ 10,\ 10}$	50	1,000	5	53
partite3 = $K_{5,\ 5,\ 5,\ 5,\ 5,\ 5,\ 5,\ 5,\ 5,\ 5}$	50	1,125	10	58

〈Table 3〉 Results obtained from the graph partite1: the best and average value of 10 executions

		PMX						OX						CX					
	λ	20		60		250		20		60		250		20		60		250	
	$p_m \backslash p_c$	0.6	0.8	0.6	0.8	0.6	0.8	0.6	0.8	0.6	0.8	0.6	0.8	0.6	0.8	0.6	0.8	0.6	0.8
DM	0.01	33 33.9	33 34.2	34 35.3	34 35.6	34 35.3	35 36.4	33 33.7	33 34.0	33 34.9	33 35.1	34 35.3	33 35.8	41 42.9	38 42.5	40 41.4	41 42.2	39 41.9	40 42.0
	0.05	33 34.0	33 34.9	33 34.6	34 35.5	34 35.3	34 36.0	33 33.8	33 34.4	33 34.5	33 35.1	33 34.7	34 35.3	40 42.3	40 42.0	38 41.4	40 42.2	41 42.2	39 41.8
	0.1	33 33.8	33 34.4	33 34.5	33 34.8	33 34.3	33 36.0	33 33.7	33 34.3	33 34.3	33 35.2	33 34.3	34 35.5	39 42.0	40 42.2	40 41.5	39 41.8	41 41.4	40 41.2
EM	0.01	34 36.1	34 36.4	33 35.4	33 35.3	35 37.7	35 37.4	33 34.7	34 34.8	33 35.9	34 36.0	34 37.0	35 36.8	34 36.7	34 36.5	34 35.8	34 35.7	33 34.0	33 34.7
	0.05	35 36.2	35 35.9	34 35.9	33 36.2	36 37.7	36 37.1	33 34.3	34 35.6	35 36.0	34 35.3	35 36.2	34 36.1	35 36.5	34 36.3	34 35.6	34 35.3	33 34.1	33 34.3
	0.1	33 35.2	33 34.6	34 36.6	36 37.0	36 37.2	36 37.4	33 35.7	33 35.9	34 36.1	35 36.2	35 37.1	35 36.4	36 37.5	33 36.5	33 35.3	35 35.5	33 34.5	33 34.2
IVM	0.01	33 33.9	33 34.1	33 34.9	34 35.7	34 35.3	35 35.9	33 33.8	33 34.8	34 34.8	34 36.0	33 34.6	33 35.6	33 33.2	33 33.0	33 33.0	33 33.1	33 33.1	33 33.0
	0.05	33 33.4	33 34.5	33 34.5	34 35.9	33 34.5	33 35.4	33 34.1	33 34.5	33 34.6	33 35.3	33 34.7	33 35.2	33 33.1	33 33.2	33 33.0	33 33.0	33 33.0	33 33.0
	0.1	33 33.4	33 33.4	33 33.9	34 35.5	33 34.2	34 35.4	33 33.6	33 34.3	33 34.3	33 34.1	33 34.3	33 35.3	33 33.0	33 33.0	33 33.0	33 33.0	33 33.0	33 33.0
SM	0.01	34 36.3	34 36.0	33 36.3	35 36.4	35 37.3	36 37.9	33 34.9	33 35.1	33 36.3	34 36.5	35 36.8	35 36.6	36 38.2	36 38.3	35 37.2	34 36.3	34 35.3	33 35.4
	0.05	33 35.5	34 35.3	34 36.2	34 36.7	35 37.4	36 37.3	34 34.9	33 34.4	34 35.9	36 37.5	34 36.8	36 36.7	37 37.9	37 38.6	34 36.3	35 37.4	33 35.1	34 35.5
	0.1	34 36.2	34 35.3	36 37.4	35 36.5	36 37.2	36 37.6	33 35.3	33 35.3	33 35.8	33 37.0	36 37.3	36 37.6	35 38.0	36 37.9	34 37.1	35 36.9	33 35.0	33 35.3
ISM	0.01	33 35.1	34 34.9	34 36.3	33 36.4	35 36.7	34 36.3	33 34.1	33 34.3	33 35.2	33 35.0	34 36.2	34 37.3	34 35.7	34 35.9	33 34.7	33 34.3	33 33.1	33 33.2
	0.05	33 35.3	34 35.7	35 36.4	34 36.0	35 36.5	35 36.2	34 34.7	34 34.8	33 35.1	34 35.9	34 35.1	34 36.3	34 35.9	33 35.3	33 34.4	34 34.6	33 33.2	33 33.1
	0.1	33 35.3	33 35.5	35 36.2	35 36.4	34 35.9	35 36.7	33 34.3	34 35.0	34 35.1	34 35.5	34 36.4	34 36.5	33 34.8	33 35.4	33 34.5	33 34.9	33 33.3	33 33.5
SIM	0.01	33 33.5	33 33.4	33 34.1	33 35.7	33 35.5	35 36.6	33 33.7	33 34.7	33 34.2	34 35.5	33 34.7	34 36.0	33 33.0	33 33.0	33 33.0	33 33.0	33 33.0	33 33.0
	0.05	33 33.2	33 33.2	33 33.7	33 35.0	33 34.1	33 35.5	33 33.6	33 34.3	33 33.9	33 34.9	33 34.2	34 35.6	33 33.0	33 33.0	33 33.0	33 33.0	33 33.0	33 33.0
	0.1	33 33.3	33 33.2	33 33.8	34 35.4	33 33.6	33 35.5	33 33.2	33 33.5	33 33.1	33 34.4	33 33.5	33 34.7	33 33.0	33 33.0	33 33.0	33 33.0	33 33.0	33 33.0

Theorem 1. [6] Let $G = (V, E)$ be a complete k-partite graph with $|V| = n$. Then $\lambda(G) = n + k - 2$.

(Fig. 3) when applied to complete k-partite graphs yield optimum $L(2,1)$-labelings.

```
Algorithm Opt
Input: A complete k-partite graph G = (V, E) with partite
       sets {p1, p2, ..., pk} where k ≥ 2.
Output: Optimum L(2,1)-labeling of G with λ(G) = |V|+k - 2.
begin
    t = 0;
    for i = 1 to k do
        for j = 1 to |pi| do
            select unlabeled vertex v from pi;
            label v by t;
            t = t + 1;
        end
        t = t + 1;
    end
end
```

(Fig. 3) Optimum L(2,1)-labeling algorithm for the complete k-partite graphs

Algorithm *Opt* suggests that, in order to achieve the optimality for the $L(2,1)$-labelings of complete k-partite graphs, each partite set must consume the consecutive labels. Therefore, the complexity of finding optimum $L(2,1)$-labeling of complete k-partite graphs depends on the number of partite sets.

For the evaluation of our genetic algorithms, we consider three different population size λ (λ = 20, 60 and 250), two different crossover rates p_c (p_c = 0.6 and 0.8), and three different mutation rates p_m (p_m = 0.01, 0.05 and 0.1). For the termination condition we applied fixed number of 2,000 iterations. <Table 3, 4> and <Table 5> contain the results of applying our GA to the three graphs shown in <Table 2> for all possible combinations of genetic operators mentioned above; the best and average value of 10 executions of the algorithms.

From <Table 3> we see that for the graph *partite*1 all combinations of genetic operators except CX + DM found the optimum value of 33. For the combination of CX + SIM, regardless of population size and crossover and mutation rate the algorithm found the optimum value of *partite*1. Therefore, CX + IVM and CX + SIM are the

〈Table 4〉 Results obtained from the graph partite2: the best and average value of 10 executions

		PMX						OX						CX					
	λ	20		60		250		20		60		250		20		60		250	
	$p_m \backslash p_c$	0.6	0.8	0.6	0.8	0.6	0.8	0.6	0.8	0.6	0.8	0.6	0.8	0.6	0.8	0.6	0.8	0.6	0.8
DM	0.01	56 59.3	57 59.7	59 60.7	62 63.6	59 62.1	60 63.3	56 58.7	57 58.6	56 59.6	59 62.0	58 61.1	62 64.3	54 55.9	54 55.8	53 54.5	53 54.7	53 53.6	53 53.8
	0.05	55 58.2	56 58.3	56 59.0	60 62.5	58 60.1	60 63.5	57 57.9	56 59.4	57 60.2	58 60.8	56 61.3	59 62.2	54 55.3	54 55.7	53 54.3	53 54.5	53 53.9	53 53.6
	0.1	55 57.3	57 59.0	56 59.0	60 61.9	56 59.5	59 63.3	55 57.0	57 59.0	55 58.1	59 60.9	58 61.0	59 62.2	54 55.2	54 54.8	54 55.2	54 54.5	53 54.7	53 54.2
EM	0.01	59 60.4	59 61.1	60 62.4	62 63.7	63 64.9	62 66.3	59 61.4	59 60.4	61 63.6	63 65.1	61 65.1	62 64.8	59 62.2	59 62.1	58 60.4	58 61.1	58 59.6	57 58.6
	0.05	57 60.8	58 59.9	58 61.2	58 62.8	60 64.0	61 64.9	56 61.0	59 61.1	58 61.9	58 63.1	61 64.6	60 64.8	59 62.4	58 61.8	57 61.8	58 61.1	57 58.7	56 57.8
	0.1	59 61.1	61 62.1	59 62.6	61 63.6	61 64.5	64 66.0	58 60.6	60 61.5	58 62.3	61 63.9	62 64.6	63 65.4	56 61.8	61 62.0	58 61.3	58 60.7	56 58.9	57 58.5
IVM	0.01	57 58.6	56 59.2	59 60.7	59 62.4	60 62.3	60 63.3	57 59.0	55 59.5	58 60.2	59 61.9	59 61.7	59 64.1	53 54.5	53 55.3	53 53.9	53 54.2	53 53.8	53 53.7
	0.05	57 58.3	58 59.6	55 58.2	61 62.7	56 59.3	63 64.9	55 56.6	56 59.5	55 59.5	57 61.0	57 60.8	60 62.3	53 54.9	53 55.2	53 54.3	53 53.8	53 53.9	53 53.7
	0.1	55 56.8	56 58.9	57 60.3	59 61.3	56 59.4	60 62.0	55 57.3	56 59.7	54 57.5	59 61.2	57 60.5	57 62.0	53 54.3	53 54.8	53 54.0	53 54.2	53 53.9	53 53.8
SM	0.01	58 62.4	58 60.3	63 64.5	61 63.7	63 65.2	62 65.0	58 61.2	57 60.8	61 65.0	60 63.9	60 65.3	64 66.2	63 68.0	61 66.1	60 65.0	63 64.5	56 60.5	59 60.7
	0.05	59 61.2	58 61.7	62 64.8	62 65.1	63 66.3	65 66.7	58 61.4	62 63.0	62 65.0	62 64.1	64 65.9	62 65.5	64 65.9	63 65.5	60 64.0	62 64.7	59 61.3	58 62.5
	0.1	60 62.2	60 62.3	63 65.7	61 64.4	62 65.3	60 64.4	61 63.0	60 62.5	61 64.9	62 65.6	63 66.0	64 66.2	63 66.0	63 66.4	63 65.0	59 64.2	59 63.1	61 63.1
ISM	0.01	59 60.4	58 60.4	58 61.9	61 63.4	60 63.8	60 64.4	56 59.5	59 60.8	59 61.7	60 62.9	61 63.9	64 66.2	58 62.2	60 61.9	58 61.2	58 60.1	54 58.1	53 57.1
	0.05	56 59.3	56 60.0	60 62.2	60 63.4	61 64.3	62 64.9	57 58.7	58 60.1	58 61.1	58 62.8	59 61.9	58 62.8	58 61.9	57 59.8	59 61.3	57 60.7	55 57.6	55 57.5
	0.1	58 60.7	58 60.6	60 61.7	61 63.5	59 61.7	60 62.8	56 59.3	58 60.3	60 62.4	58 62.5	58 61.6	61 64.3	59 61.2	60 62.4	59 60.7	58 61.8	58 60.2	55 58.0
SIM	0.01	54 56.5	55 57.9	56 60.5	57 61.9	58 61.5	62 64.5	56 58.5	57 59.4	57 60.3	60 63.2	58 60.8	60 62.9	53 54.6	53 54.2	53 54.3	53 53.5	53 53.3	53 53.3
	0.05	56 57.0	55 57.4	56 59.9	60 62.6	57 60.3	56 62.5	55 56.9	57 59.4	58 59.9	60 61.7	57 60.3	60 61.9	53 54.2	53 54.2	53 53.7	53 53.5	53 53.4	53 53.7
	0.1	54 56.8	55 58.5	55 58.4	60 61.9	56 59.2	59 61.7	55 56.9	58 59.3	55 58.1	56 61.1	56 58.4	60 61.1	53 53.8	53 53.7	53 53.5	53 53.9	53 54.1	53 54.3

most effective combination of genetic operators for partite1.

From <Table 4> even though CX + DM found optimum value when $\lambda \geq 60$, CX + IVM and CX + SIM found optimum value of 53 in all cases.

<Table 5> also shows that for the graph *partite3*, CX + SIM and CX + IVM are the two most effetive combination of genetic operators for this problem.

〈Table 5〉 Results obtained from the graph partite3: the best and average value of 10 executions

		PMX						OX						CX					
	λ	20		60		250		20		60		250		20		60		250	
	$p_m \backslash p_c$	0.6	0.8	0.6	0.8	0.6	0.8	0.6	0.8	0.6	0.8	0.6	0.8	0.6	0.8	0.6	0.8	0.6	0.8
DM	0.01	63 65.9	64 67.8	65 68.6	69 71.9	68 71.1	70 71.6	64 66.6	67 68.3	66 69.5	67 71.2	68 70.9	70 72.0	81 84.8	81 83.9	80 83.3	80 82.7	80 82.9	80 83.3
	0.05	63 66.1	66 67.3	63 68.3	68 71.8	67 70.5	70 71.9	62 65.5	64 66.6	66 68.5	69 71.4	67 69.8	70 72.6	81 84.1	81 83.3	78 82.5	78 81.1	82 83.3	81 82.9
	0.1	61 64.8	64 66.3	67 68.3	68 69.9	67 68.8	68 70.6	64 66.1	62 66.6	63 66.5	66 69.0	64 67.2	67 70.6	81 83.7	79 82.7	80 82.8	79 82.7	79 82.7	80 82.2
EM	0.01	65 66.7	63 66.7	65 66.9	67 70.2	70 73.7	70 73.2	64 67.2	65 68.1	67 71.0	68 72.5	69 72.5	73 74.1	67 68.5	65 68.6	65 66.4	64 66.9	64 65.5	62 64.2
	0.05	62 66.7	64 66.4	66 68.2	68 70.8	70 72.6	68 71.9	64 66.7	65 69.4	69 71.6	66 71.4	69 71.5	68 71.9	63 67.6	65 67.5	65 67.8	61 66.3	62 64.1	61 64.0
	0.1	64 66.4	62 66.4	66 69.4	68 71.6	69 71.1	69 71.1	67 69.3	66 69.7	67 70.3	69 72.1	69 72.0	70 73.4	64 67.9	63 66.5	65 67.4	63 66.6	63 65.4	63 65.4
IVM	0.01	66 67.3	60 67.2	65 68.9	67 70.7	67 70.1	69 71.7	64 66.1	64 66.9	65 69.8	68 71.1	67 70.5	67 71.9	60 62.5	60 62.7	58 60.9	58 61.1	59 60.6	58 60.3
	0.05	62 65.3	64 67.6	66 67.5	66 69.8	66 69.6	67 71.8	64 66.1	66 67.7	65 67.5	66 70.3	65 69.7	71 72.5	59 61.5	59 61.8	59 60.8	60 61.5	58 60.1	59 61.2
	0.1	64 65.2	64 67.8	65 67.4	66 70.5	65 67.9	67 70.4	63 64.9	64 67.8	64 66.3	67 69.9	62 66.9	69 71.9	60 61.2	59 60.7	59 61.4	60 61.3	58 60.3	58 60.2
SM	0.01	67 68.6	65 67.2	68 72.2	71 73.9	69 73.3	72 74.3	67 70.8	68 69.6	71 73.5	71 73.3	72 74.6	72 74.9	73 76.8	72 75.5	71 73.6	70 73.2	70 72.0	67 69.2
	0.05	67 69.4	65 67.9	70 73.5	70 73.4	72 74.4	73 74.5	69 70.5	68 69.6	70 73.2	72 74.3	72 75.1	71 73.9	73 74.5	73 76.3	70 72.5	71 74.1	67 70.9	67 69.9
	0.1	66 69.3	68 69.9	71 73.0	70 72.3	71 74.6	71 73.8	68 71.5	66 70.9	71 73.0	70 72.9	70 73.9	71 73.8	72 75.8	74 76.3	71 73.8	71 73.1	69 72.3	64 70.6
ISM	0.01	62 65.2	62 64.8	66 69.5	68 71.3	69 71.0	70 72.4	63 67.2	65 67.8	67 69.4	70 71.2	68 71.4	70 72.7	64 66.0	64 65.0	61 64.2	60 64.1	60 63.3	60 63.4
	0.05	64 65.9	61 66.1	66 68.5	69 71.2	68 70.5	68 72.4	64 66.5	65 67.2	66 70.5	68 71.1	67 69.7	69 73.2	63 64.0	63 65.9	63 64.7	63 64.8	62 64.1	61 64.1
	0.1	63 65.9	63 65.4	66 68.3	68 70.1	67 69.5	69 70.8	65 67.0	65 67.9	64 68.6	69 70.7	67 69.9	69 71.4	63 65.4	62 65.2	62 64.0	63 65.3	58 64.6	62 65.7
SIM	0.01	60 62.8	63 64.6	66 69.3	69 71.3	66 70.7	70 72.9	65 66.4	65 68.1	65 69.5	69 71.6	66 68.9	69 71.4	60 60.9	59 60.8	58 60.1	58 60.6	58 59.3	58 60.5
	0.05	60 63.3	63 65.7	65 67.4	67 70.5	66 68.8	69 71.0	63 65.6	64 67.5	67 68.7	67 70.0	65 68.9	68 71.0	58 60.8	58 59.3	58 59.8	58 59.6	58 59.0	58 60.4
	0.1	60 63.2	62 65.9	62 64.8	68 70.4	62 67.6	67 70.8	59 63.6	63 65.9	65 67.2	66 69.5	65 67.9	67 70.5	58 60.0	58 59.5	58 59.1	58 59.3	58 59.3	58 60.7

However, if $\lambda \leq 20$, CX + IVM failed to find optimum value while CX + SIM failed only when $p_m = 0.01$.

In overall, based on the results of our experiments, we can conclude that CX + SIM with $\lambda \geq 60$ is the best combination of genetic operators for solving the $L(2,1)$-labeling of graphs with genetic algorithms.

4. Conclusions

We have applied genetic algorithm to the $L(2,1)$-labeling of graphs and found very efficient combination of genetic operators. Since there are no known previous results of applying genetic algorithms to $L(2,1)$-labeling problems we applied our GA to the class of graphs with known optimum values and found that CX + SIM is the best combination of genetic operators. CX + ISM with path representation for the chromosomes our algorithm found optimum $\lambda(G)$ values for the three test graphs.

As we introduced in Section 1, $L(2,1)$-labeling is a special case of $L(j,k)$-labeling of graphs. Therefore, we will continue to investigate the applicability of genetic algorithms to the problem of $L(j,k)$-labeling problems.

References

[1] D. Goncalves, *On the L(p, 1)-labelling of graphs*, Discrete Math. and Theor. Comp. Science AE, pp. 81-86, 2005

[2] F. S. Roberts: Working group agenda of DIMACS/DIMATIA/Renyi working group on graph coloring and their generalizaitions, 2003.

[3] Gerard J. Change, Changhong Lu, *Distance-two labelings of graphs*, European Journal of Combinatorics 24 pp. 53-58, 2003

[4] Hans L. Bodlaender, Ton Kloks, Richard B. Tan and Jan van Leeuwen, *Approximations for λ-coloring of graphs*, The Computer Journal 47, pp. 193-204, 2004.

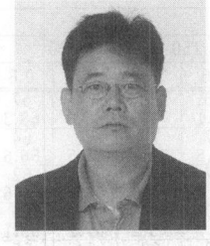
[5] Holland, J., *Adaptation in Natural and Artificial Systems*, University of Michigan Press, Ann Arbor.

[6] J. R. Griggs, R. K. Yeh, *Labelling graphs with a condition at distance two*, SIAM J. Discrete Math. 5, pp. 586-595, 1992.

[7] Oliver, I. M., Smith, D. J and Holland, J. R. C, *A study of permutation crossover operators on the TSP, In Genetic Algorithms and Their Applications*: Proceedings of the Second International Conference, Cambridge, MA, pp. 224-230.

[8] Pedro Larranaga, Cindy M.H. Kuijpers, Mikel Poza and Roberto H. Murga, *Decomposing Bayesian networks: triangulation of the moral graph with genetic algorithms*, Statistics and Computing 7, pp. 19 - 34, 1997.

[9] Zbigniew Michalewicz, *Genetic Algorithms + Data Structures = Evolution Programs*, Springer-Verlag, Berlin, 1992



한 근 희

e-mail : kehan@kongju.ac.kr

1986년 건국대학교 물리학과 졸업(학사)

1992년 Univ. of Central Oklahoma 응용수학과 졸업(이학석사)

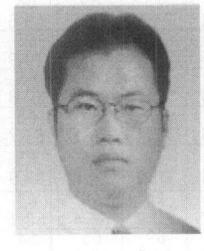
1996년 Univ. of Oklahoma 컴퓨터과학과 졸업(이학박사)

1996년~2000년 한국전자통신연구원

1999년~2000년 미국 NIST 객원연구원

2000년~현재 공주대학교 응용수학과 부교수

관심분야 : 그래프 알고리즘, Genetic Algorithm



김 찬 수

e-mail:chanskim@kongju.ac.kr

1991년 부산대학교 전산통계학과 졸업

1997년 부산대학교 통계학과 (이학박사)

2002년~현재 공주대학교 응용수학과 조교수

관심분야 : 베이지안 네트워크, 유전 알고리즘