시간표 작성 문제를 위한 가중치 그래프 착색 알고리즘에 관한 연구

김 명 재† · 안 종 일† · 정 태 충††

요 약

대학의 시간표 문제를 해결하기 위해서 시간표의 문제는 종종 그래프 착색 알고리즘으로 표현된다. 각 노드를 강좌로, 에지는 연결된 강좌간에는 동일한 시간에 서로 배정이 불가능한 조건으로 표시한다. 그러나 순수한 그래프 착색 알고리즘은 바로 대학의 시간표 문제에 적용하기 곤란하다. 그 이유는 대부분의 시간표 문제에서 강좌는 한시간 이상으로 존재하고 이들은 반드시 연속된 시간에 배정이 되어야 한다는 등의 제약 조건을 갖기 때문이다. 이 문제를 해결하기 위하여 적용된 것이 가중치 그래프 착색 알고리즘이다. 이 경우 각 노드는 강의 시간의 길이의 값을 갖게 된다. 이와 관련된 기존의 연구가 진행되었으나 탐색의 시간이 지수적으로 증가하거나 해의 질이 뛰어나지는 못하였다라는 단점을 갖고 있다. 따라서 본 연구에서는 새로운 가중치 그래프의 착색 방법을 제안한다.

A Study of Weighted Graph Coloring Algorithm for Timetabling Problem

Myung-Jae Kim† · Jong-Il Ahn† · Tae-Choong Chung††

ABSTRACT

Timetabling problems have often been formulated as coloring problems in graph. The lectures are represented by vertices, and edges join those pairs of vertices whose corresponding lectures can not be assigned in same period. Such formulation, however, imposes a significant limitation on the flexibility of the model of timetabling-problem, since it is not possible to incorporate any of the consecutive lecture constraints that frequently occur in real problem. Therefore, the concept of weighted graph coloring have been introduced to overcome this limitation. In this case, the each node has a weight number representing the length of a lecture. Many researches have been dealt with this problem but they have difficulties to overcome exponentially increased search time and to guarantee good solution. We propose a new weighted-graph-coloring algorithm to solve it.

1. 서 론

대학의 시간표 문제에서 교수, 학생, 강의실의 중복없이 스케줄링이 가능한 최소한의 시간 범위를 알아내는 것은 대단히 어렵고 복잡하다[6,7,14]. 이 문제를 해결하기 위해서 시간표의 문제는 종종 그래프 착색 알고리즘으로 표현된다[3,4,8,9,10]. 그래프 착색 문제에서는 각 노드를 강좌로, 에지는 연결된 강좌간에는 동일한 시간에 서로 배정이 불가능한 조건으로 표시한다. 따라서 최소한의 시간 범위는 착색된 색의 수로서 알 수 있다.

† 준 회 원 : 경희대학교 대학원 전자계산공학과
†† 정 회 원 : 경희대학교 전산과 교수, KAIST CAIR 연구원
논문접수 : 1998년 8월 6일, 심사완료 : 1998년 9월 23일

그러나 순수한 그래프 착색 알고리즘은 바로 대학의 시간표 문제에 적용하기 곤란하다. 그 이유는 시간표 문제에서 대부분의 강좌는 한시간 이상으로 존재하고 이들은 반드시 연속된 시간에 배정이 되어야 한다는 제약 조건을 갖기 때문이다. 이 문제를 해결하기 위하여 적용된 가중치 그래프 착색 알고리즘은 각 노드가 가중치를 갖도록 하는 것이다. 여기서 가중치는 각 강좌의 시간의 길이를 의미한다. 즉, 이 문제에서는 서로 연결된 노드 사이에 중복된 색을 갖지 않도록 하며, 각 가중치에 해당하는 수 만큼의 색을 연속적으로 부여하는 문제라 정의할 수 있다.

이와 같은 가중치 그래프 착색 알고리즘으로 시간표의 문제에 적용된 연구는 Clementson[1]과 Cangalovic[2]에 의해서 진행되었다. 이 중 Cangalovic이 제안한 알고리즘은 일종의 백트랙킹에 의한 방법으로 Clementson의 연구 결과 보다 좀 더 최적에 가까운 해를 찾을 수 있음을 보여 주고 있지만[2], 이 방법은 그래프의 크기의 증가에 따라 탐색의 시간이 지수적으로 증가하게 되고 대학 시간표와 같이 대규모의 그래프를 착색하는 방법으로서는 실용적이지 못하다.

반면 Clementson의 알고리즘은 시간표를 길이 우선순위로 적용하고 이 순서에 입각하여 착색을 수행한다. 만약 적당한 시간을 찾지 못했다면 이미 색이 배정된 노드 중 충돌이 발생하는 한 개의 노드에 한해서 다른 색으로 변경을 시도한다. 이 변경의 시도가 실패하면 현재 배정하고자 하는 노드의 색은 현재 까지 배정된 최대의 색으로 배정하고 탐색을 종결한다. 이와 같이Clementson의 재배치의 전략은 간결하고 간단한 그래프의 착색에서는 효과적이지만, 탐색의 공간을 과도하게 제약하므로 해의 질이 뛰어나지는 못하다.

그러므로 본 연구에서는 이미 배정 받은 다른 색으로의 변경의 탐색 과정을 재귀적으로 수행하는 새로운 가중치 그래프의 착색 방법을 제안한다. 이 방법은 현재 배정된 총 색의 공간에서 재배치를 통해 현재 배정하고자 하는 노드의 색을 확보한다는 의미에서 Placemaker라고 명명한다.

제 2장에서는 Clementson의 알고리즘을 자세히 소개하고 3장에서 이 방법의 문제점과 새로운 착색 방법을 제안한다. 4장에서는 지금 까지 제안된 가중치 그래프 착색 알고리즘들의 해의 질과 탐색 시간과의 비교 실험을 수행한다. 마지막 5장에서 결론 및 향후 연구의 방향을 기술하고 결론을 맺는다.

2. 가중치 그래프 착색

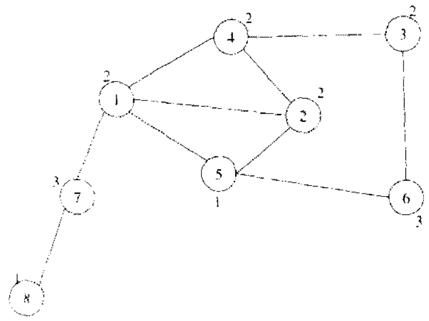
2.1 그래프의 정의

가중치 그래프 G는 (V(G), E(G), C(V(G)))와 같이 정의할 수 있다. V(G)는 그래프의 노드의 집합이며, E(G)는 에지의 집합을 말한다. 또한 C(V(G))는 노드에 착색 되어야 할 색의 수이다. 즉, 노드의 집합 $\{v_i\}$에서 각 노드는 필요한 색의 수 c_i^v 를 갖게 된다. n개의 노드 간의 에지의 연결 관계는 n x n으로 구성된 2차원 행렬 A =$[a_{ij}]$으로 표시된다. 이 때 a_{ij} =0 또는 1로서 노드 i와 j는 서로 연결관계를 설정한다.

노드 v_i는 색상도 f_i라는 값을 갖게 되는데 그것은 식 (1)과 같이 정의할 수 있다.

$$f_i = \sum_{j=1}^{n} a_{ij}\, c_j^v + c_i^v \qquad (1)$$

이 색상도 f_i는 노드의 착색 순서를 결정할 때 사용하는 값이 된다.



(그림 1) 가중치 그래프의 예
(Fig. 1) The example of weighted graph

〈표 1〉 그림 1 그래프의 색 배정 예
〈Table 1〉 The result of fig. 1 graph

노드	색의 수	배정된 색
1	2	5,6
2	2	3,4
3	2	4,5
4	2	1,2
5	1	7
6	3	1,2,3
7	3	1,2,3
8	1	4

2.2 가중치 그래프 착색과 시간표 문제

시간표의 문제를 최대한 간단히 표현하면 다음과 같다. 시간표 문제란 1시간 이상의 시간이 필요한 강좌에 담당 교수의 시간과 학생의 시간이 중복되지 않고 적당한 시간을 배정하는 문제라 정의할 수 있다. 시간표의 문제에서 각 노드는 강좌에 해당하며, 강좌간에 에지로 연결은 강좌간에 충돌이 발생하는 것을 말한다.

〈표 2〉 가중치 그래프와 시간표 문제의 비교
〈Table 2〉 The comparison of weighted graph and Timetable Problam

시 간 표	가중치 그래프
과목	노드
과목간에 충돌	에지
과목의 시간 길이	가중치
전체 시간표의 시간길이	사용된 전체 색의 수

그래프에서 노드의 가중치는 강좌의 시간 길이가 된다. 그래프의 착색의 조건은 각 노드가 가중치에 해당하는 만큼의 연속된 시간을 에지로 연결된 다른 강좌와 충돌 없이 배정하는 것을 말한다.

따라서 가중치 그래프 착색을 통해 각 강좌가 어떤 시간에 배정되었을 때 최소의 시간표 길이가 될지를 알 수 있다.

각 강좌 x_i에 할당된 색을 $c(x_i)$라고 하고 해당 강좌의 가중치가 $w(x_i)$일 때, 이것을 목적 함수로 표현하면 식(2)와 같다.

$$\text{minimize} \quad \max_{i=0}^{n}(c(x_i)+w(x_i)) \tag{2}$$

2.3 노드의 탐색 순서

NP hard로 분류되는 그래프 착색 문제의 대부분 탐색(exhaustive search)에 의해 탐색하는 것은 탐색의 공간과 시간의 문제로 인해 불가능하다. 따라서 대부분의 휴리스틱 방법에서는 First Fit그래프 착색 알고리즘에 근거한 방법을 사용한다. First Fit알고리즘은 충돌이 없는 가장 작은 색을 노드에 착색하는 방법이다. 따라서, 이 휴리스틱의 핵심은 착색할 노드의 순서이다. 가중치 그래프 착색 알고리즘에서 사용하는 대표적인 노드의 정렬 방법은 다음 두 가지가 있다.

LF1 : 가장 가중치가 높은 노드를 우선으로 착색한다. 만약 동일한 경우 색상도 1값이 높은 것을 우선으로 한다.

LF2 : 가장 색상도가 높은 노드를 우선으로 착색한다. 만약 동일한 경우 가중치가 높은 것을 우선으로 한다.

위의 노드 정렬 방법에 대한 실험은 Clementson[1]과 Cangalovic[2]에 의해서 수행되었다. 이 두 논문에서 모두 LF1에 의한 방법이 더 좋은 결과를 보임을 실험을 통해 밝히고 있다. 본 논문에서도 역시 LF1에 의한 탐색 방법을 사용한다.

2.4 최소 착색의 수의 예측

하나의 그래프에서 착색을 위해 필요한 최소의 색의 수를 미리 예측 하는 것은 매우 중요하다. 이 예측치는 탐색의 공간을 제어하는데 사용될 수 있기 때문이다. 일반 그래프에서의 최소 착색을 위한 대표적인 방법은 전체 그래프 내에서 가장 큰 완전 그래프(complete graph)를 찾아 내는 방법이다. 가중치 그래프에서 역시 완전 그래프를 기반으로 한 최소 착색 수를 결정하는 방법을 사용하지만, 완전 그래프의 크기는 완전 그래프 내의 그래프가 갖는 가중치의 합으로 결정된다. 그래프 내의 완전 그래프를 H라고 하면 이의 식은 (3)과 같다.

$$\max_{H} \sum_{v_i \in H} w(v_i) \tag{3}$$

3. Clementson가중치 그래프 알고리즘

색의 집합이 $1,2,\ldots,C_k$이고 노드가 v_1,v_2,v_3으로 정렬되어 있을 때 그래프의 착색 방법으로 가장 간단한 것은 순서에 따라 차례로 충돌이 없는 최소의 색으로 착색해 나가는 것이다. 이 방법에서 중요한 것은 어떠한 순서로 노드를 착색 할 것인가 이다. 가중치 그래프에서의 착색의 순서는 가중치가 높은 것을 우선으로 둥들인 경우 색상도(chromatic degree)가 높은 것을 먼저 착색하는 방법을 사용한다(Largest First: LF). 즉, 시간표의 관점에서는 시간 길이가 긴 강좌를 우선으로 그리고 동일한 시간 길이를 갖는 것 중에서는 다른 강좌와의 중복될 가능성이 높은 강좌를 우선으로 착색하는 것이다.

Clementson의 착색 방법은 위의 순서에 입각한 노드 착색 방법과 이미 착색되어 있는 노드의 색을 변경하는 부분 탐색의 휴리스틱을 사용한 것이다. 방법의 전체 프로시저 WGC(Weighted Graph Coloring)를 정의하면 다음과 같다.

1. 노드를 LF정렬 방법에 따라 정렬한다. 정렬된 노드의 순열을 <$v_1, v_2, \ldots, v_n$>이라 한다.
   또한 색의 집합은 <$1, 2, \ldots, c_k$>이라고 하자.
2. 첫번째 노드에 색1에서부터 가중치 c_1^v에 해당하는 값 만큼을 연속적으로 부여하고 변수 MAX = c_1^v으로 설정한다. MAX는 현재 까지 착색된 색의 최대값을 말한다.
3. 다음 노드를 결정하고 노드에 색 $1, 2, .., c_k$중 이미 착색된 노드와 충돌이 없는 가장 작은 색j가 MAX >= j+ c_i^v 1이라면 착색한다.
4. 만약 MAX < j+ c_i^v -1 이면, 색의 집합의 부분 집합인 P {P_k}를 구한다. 집합 P는 다음과 같이 정의될 수 있다. 만약 노드 v_i가 색 P_k으로 착색된다면, 이미 착색된 노드 중 한 개와 충돌이 발생하는 색.
5. 만약 집합 P=Ø 이라면 노드 v_i에 색j를 할당하고 MAX = j+ c_i^v -1로 한다.
6. 그렇지 않다면, P_k에 노드v_i 가 착색 됨으로써 충돌이 발생하는 노드 v_d가 MAX범위 내에서 다른 색으로 착색될 수 있는지를 탐색한다.
7. 만약 탐색에 실패하면, 노드 v_i에 색j를 할당하고 MAX = j+ c_i^v -1로 한다.
8. 탐색이 성공한다면, 다시 착색된 노드 v_d의 색과 현재 노드 v_i에 할당된 색이 최단인 자리에 노드 v_i의 색 P_k를 할당한다.

4. Place-Maker 가중치 그래프 착색 알고리즘

Clementson의 탐색 방법의 특징은 전체 프로시저의 단계 6에서 단계 8까지의 이미 배정 받은 노드의 색을 다른 색으로 배정하는 것이다. 이러한 간단한 방법은 그래프의 복잡도가 낮은 환경에서는 효과적일 수 있지만, 실제 문제에서는 그 효과를 발휘하기가 쉽지 않다.

본 논문에서는 이미 배정 받은 노드의 색을 다른 색으로의 변경이 실패하는 경우 재귀적인 방법으로 탐색의 트리를 확장 함으로서 효과를 극대화 시키는 방법을 사용한다. 이것에 대한 전체 프로시저는 다음과 같다.

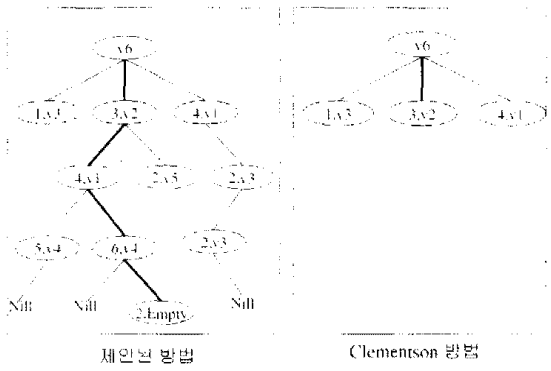
1. 노드를 LF정렬 방법에 따라 정렬한다. 정렬된 노드의 순열을 <$v_1, v_2, \ldots, v_n$>이라 한다.
2. 첫번째 노드에 색1에서부터 가중치 c_1^v에 해당하는 값 만큼을 연속적으로 부여하고 변수 MAX = c_1^v으로 설정한다. MAX는 현재 까지 착색된 색의 최대값을 말한다.
3. 현재의 노드v_c ← v_i로 설정한다 만약 i>=n.이면 종료
4. 집합 P ← Ø와 P_{visit} ← Ø로 초기화 한다.
5. 노드 v_c에 색 $1, 2, \ldots, c_k$중 이미 착색된 노드와 충돌이 없는 색j가 MAX >= j+ c_c^v 1이라면 착색하고 만약 P_{visit} ≠ Ø이라면 다음을 수행한다.
   - 5-1 P_{visit}집합으로부터 $P' = \{(P'_k, v'_k, v'_c) \mid P_{visit}(P'_k, v'_k, v'_c), v_c = v'_k\}$를 구한다.
   - 5-2 P'= Ø이면 단계 3으로
   - 5-3 v_c' 에 P'_k를 착색한다.
   - 5-4 v_c ← v'_c로 하고 단계 5-1로
6. 만약 MAX < j+ c_c^v -1 이면, 노드 v_c가 색 P_k으로 착색된다면, 이미 착색된 노드v_k와 충돌이 발생한다는 의미의 튜플 (P_k, v_k, v_c)을 모두 구하고 집합 $P = \{(P_k, v_k, v_c)\}$에 추가한다.
7. 만약 집합 P= Ø이라면 단계 9로.
8. 집합 P로부터 하나의 원소 (P_k, v_k, v_c)를 선택하고 P로부터 제거한 후 P_{visit}집합에 삽입하고 v_c← v_k로 하고 단계 5로
9. v_c에 색 MAX를 할당하고 MAX←MAX+ c_c^v-1로 한다.
10. 단계 3으로

위의 과정은 일종의 넓이 우선 탐색에 해당한다. 프로시저는 초기에 노드를 깊이 우선 순위로 그리고 동률일 경우에는 색상도가 높은 것을 우선으로 정인하여 착색의 순서를 결정한다. 착색된 첫번째 노드에 색의 집합으로부터 가장 작은 색을 부여하고 지금까지 사용된 최대의 색의 수 MAX변수는 현재 할당된 노드의 색의 수 즉 가중치가 된다. 탐색 과정에서 만약 이 MAX변수 내에서 현재 할당될 노드의 색이 착색 가능하다면 착색을 수행하지만, 그렇지 못한 경우에는 넓이 우선 탐색을 시작한다. 단계 5에서 8단계가 그것이다. 우선 현재 착색하고자 하는 노드 v_c와 이미 색이 배정된 노드 v_k간에 충돌이 한 노드와 발생하는 색에 한하여 탐색을 시작한다. 현재 배정될 노드의 색을 충

동한 노드의 색 P_k으로 대치하고, 충돌한 노드들을 현재 탐색 되어야 할 노드로 설정한다. 즉, (P_k, v_m, v_n)는 노드 v_i가 노드 v_n에 배정된 색 P_k를 할당 받고 노드v_m을 재 탐색하는 것을 말한다.

이 과정을 반복 수행하고 모든 사례에서도 탐색이 실패한다면, 지금까지 착색되지 않은 색으로 노드를 착색한다.

위의 과정을 탐색 트리로 표현한 예는 다음과 같다.



(그림 2) 제안된 방법과 Clementson방법의 탐색 범위의 비교 예
(Fig. 2) The search spaces of proposed Method and Clementson Method

그림 2의 예는 최초 v_1,.....,v_5까지 착색이 되어 있는 상황에서 v_6를 착색하는 예를 보이고 있다. v_6는 이미 착색된 색 범위 내에서 착색이 불가능하여 이미 배정된 색을 대치하여 배정하는 탐색을 시작한다. 탐색 결과 해의 경로는 v_6 (3,v_2)- (4,v_1) (6,v_4)- (2,empty)이다. 이는 v_6는 v_2에 착색된 색 3으로 착색이 되고 v_2는 v_1의 4에, v_1은 v_4의 6에 각각 대치되고 마지막 v_4는 색2로 충돌 없이 배정됨을 의미한다.

반면 Clementson의 방법은 제안된 방법의 탐색 공간 중 탐색 트리의 첫번째 길이 만큼을 탐색하는 것으로 해의 질이 제안된 방법에 비해 좋지 않다.

5. 실 험

그래프를 생성하기 위해서 노드의 개수n과 에지의 연결도(graph density)p를 정의하고 각 노드가 갖을 수 있는 가중치의 확률 q를 정의한다. n개의 노드로 구성된 그래프의 경우 생성 가능한 최대의 에지의 개수는 n(n-1)/2개가 된다. 연결도 p는 0<p<1의 확률로 에지를 생성한다. 다음은 가중치의 확률로 이 값은 다음 확률 변수의 식(4)로 설정한다[1][2].

$$P(c_i = k) = \frac{q^k}{(e^q - 1)k!} \qquad k = 1, 2, \ldots . \qquad (4)$$

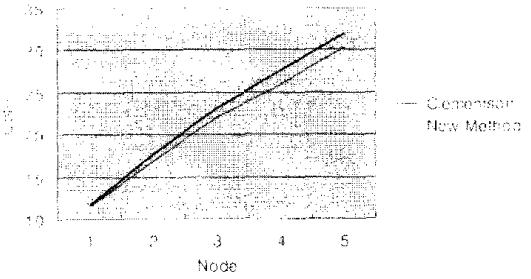
여기서 q는 1이상의 임의의 실수로서 거의 실험 결과에 영향을 미치지 않고 있음을 [2]논문에서 밝히고 있다. 본 논문에서는 q의 값은 1로 설정한다.

실험은 그래프의 노드를 20, 40,.....100개의 5가지 유형과, 각각의 노드 수에 대해 p의 값은 0.2, 0.3,.....,0.8까지 7가지 유형의 총 35개의 그래프에서 각각 20회의 반복 측정을 통해 결과를 산출한다. 각 결과는 Clementson의 결과와 비교한다.

〈표 3〉 실험 결과
〈Teble 3〉 The table of total result

P	Test	#Node	Average Clementson (Cl)	Average Prosposed Method(New)	Cl New
0.2	20	20	6.4	6.4	0
		40	8.8	8.2	0.6
		60	12	11.2	0.8
		80	13.2	12.2	1
		100	14.6	14.2	0.4
0.4	20	20	9.6	9	0.6
		40	13	12.4	0.6
		60	18.4	16.6	1.8
		80	21	20	1
		100	24.4	23	1.4
0.6	20	20	12.6	12.6	0
		40	20	18.6	1.4
		60	25.6	24.6	1
		80	31	29.2	1.8
		100	35.4	33.8	1.6
0.8	20	20	18	18	0
		40	29	28	1
		60	37	36.2	0.8
		80	45	42.6	2.4
		100	53.4	50.4	3

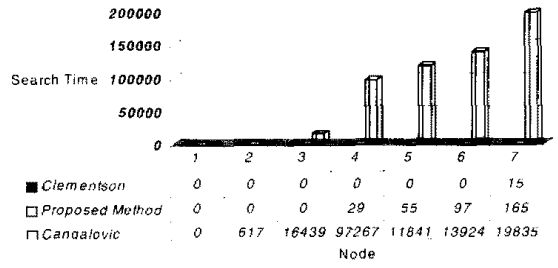
실험 결과 본 논문에서 제안한 방법이 기존의 Clementson의 방법에 비해 좋은 해를 탐색 할 수 있음을 알 수 있다. 이 비교는 특히, 그래프의 크기가 커지고 복잡도가 높은 수록 더 좋은 결과를 내고 있다.



(그림 3) 노드의 변화 단계에 따른 착색의 수 변화
(Fig. 3) The numer of color and node

그림 3의 결과는 위의 표 3의 결과를 노드의 수에 따른 각각의 p 값에 따른 착색 결과를 평균하여 Clementson의 결과와 비교한 것이다. 즉, 노드의 개수 1단계(20)에서는 두 방법의 결과의 차이가 거의 보이지 않지만, 5단계에서는 약 2개의 차이를 보이고 있다.

다음은 본 논문에서 제안한 알고리즘과 Clementson[1]과 Cangalovic[2]이 제안한 알고리즘과의 탐색 시간을 비교한다. 그래프는 p=0.3의 복잡도에 10, 15, 20, ..., 40개의 노드를 갖는 총 7개의 그래프를 무작위로 선택하여 각각을 20회 반복 측정하여 평균을 내었다. 탐색에 사용된 시스템은 IBM-PC호환 기종으로 CPU는 펜티엄 120Mhz을 사용하였다. 탐색 시간의 값은 m sec에 해당한다.



(그림 4) 각 알고리즘의 탐색 시간의 비교
(Fig. 4) The comparison of search time

그림 4의 실험 결과에서 볼 수 있듯이 탐색 공간을 많이 제약한 Clementson의 알고리즘이 가장 작은 시간이 소모된 것에 반해 Cangalovic의 알고리즘은 벡트레킹의 방법을 사용한 것으로 탐색 시간이 노드의 수의 증가에 따라 지수적으로 늘고 있음을 알 수 있다. 따라서 Cangalovic의 방법은 비록 작은 그래프에서는 다른 알고리즘에 비해 좋은 해를 찾을 수 있지만 실제 환경에 도입하여 사용하는 것은 불가능해 보인다. 반면 제안된 알고리즘은 노드에 수 n에 대해 약 n^2의 수로 탐색 시간이 증가 함을 보여 탐색 시간에서도 적당함을 보인다.

6. 결 론

지금 까지 가중치 그래프의 새로운 착색 알고리즘을 제안하였다. 가중치 그래프 착색의 문제의 응용에 대표적인 것은 시간표의 시간 길이가 다양한 강좌의 배정에서 사용된다. 시간표 문제는 상호 동일한 시간에 배정이 불가능한 강좌를 피해 강좌를 배정하는 문제로 동일한 시간에 배정이 불가능한 조건을 그래프의 에지로 표현한다. 그러나 강좌는 최소 1시간이상으로 구성되어 있고 그 형태도 여러 가지 일 수 있다. 이러한 시간 길이의 다양성을 그래프에서는 가중치로 표현하였다. 기존에 제안된 가중치 그래프 착색 알고리즘들은 탐색의 공간을 지나치게 협소하게 제한 함으로 인하여 최종적인 해의 질이 우수하지 못하거나, 탐색의 공간을 벡트레킹 하여 전역 최적을 찾도록 노력 함으로서 탐색의 공간과 시간이 폭발적으로 늘어나는 문제를 갖고 있었다. 본 논문에서는 이 둘 간의 문제점을 보안하여 새로운 방법을 제안하였다. 제안된 방법은 이미 배정된 노드의 착색 결과 범위 내에서 넓이 우선 탐색을 이용하여 새로 착색될 노드의 공간을 확보하는 방법을 사용한다.

실험 결과 탐색 공간을 제한한 알고리즘인 Clementson에 비해 우수한 결과를 보였다.

참 고 문 헌

[1] A.T.Clementson and C.H.Elphick, "Approximate Colouring Algorithms for Composite Graphs," Journal of Operational Research Society 34/6, (1983), pp.503-509.

[2] Mirjana Cangalovic, Jan A. M. Schreuder, "Exact coloring algorithm for weighted graphs applied to timetabling problems with lectures of different lengths," European Journal of Operational Research 51, (1991), pp.248-258.

[3] Carter M.W., "Recent Development in Practical Examination Timetabling," Practice and Theory of Automated Timetabling, Lecture Notes in

Computer Science 1153, Springer Verlag, (1996), pp.3-21.

[4] Jacques Ferland, "Generalized Assignment-Type Problems: A Powerful Modeling Scheme," Proceedings of the 2nd international Conference on the Practice and Theory of Automated Timetabling, (1997) pp.27-54.

[5] Wilhelm Erben and Jurgen Keppler, "A Genetic Algorithm Solving a Weekly Course-Timetabling Problem," Practice and Theory of Automated Timetabling, Lecture Notes in Computer Science 1153, Springer Verlag, (1996), pp.198-211.

[6] Burke E.K., "Examination Timetabling in British Universities-A Survey," Practice and Theory of Automated Timetabling, Lecture Notes in Computer Science 1153, Springer Verlag, (1996), pp. 76-90

[7] Bernd Bullnheimer, "An Examination Scheduling Model to Maximize Students' Study Time," Proceedings of the 2nd international Conference on the Practice and Theory of Automated Timetabling, (1997) pp.81-97.

[8] Krarup J., D. de Werra, "Chromatic Optimization: Limitation, Objectives, Uses, References," European Journal of Operational research 11 (1982), pp.1-19.

[9] Burke E.K., Elliman D.G., "A University Timetabling System based on Graph coloring and Constraint Manipulation," Journal of research on computing in Education, Vol.27, No.1, (1993), pp.1-18.

[10] Welsh and Powell, "An Upper Bound for the Chromatic Number of a Graph and Its Application to Timetabling Problems," Computer Journal 10, No.1, (1967), pp.85-86.

[11] Georg Lawton, "Genetic Algorithms for Scheduling Optimization," AI Expert, (1992), pp.23-29.

[12] Patric Henry Winston, "Genetic Algorithms" Artificial Intelligence 3rd," Addison-Wesley publishing Company," (1992).

[13] Wilhelm Erben and Jurgen Keppler, "A Genetic Algorithm Solving a Weekly Course-Timetabling Problem," Practice and Theory of Automated Timetabling, Lecture Notes in Computer Science 1153, Springer Verlag, (1996), pp.198-211.

[14] Tim B. Cooper and Jeffrey H. Kingston, " The Complexity of Timetable Construction Problems," Practice and Theory of Automated Timetabling, Lecture Notes in Computer Science 1153, Springer Verlag, (1996), pp.284-295.

김 명 재

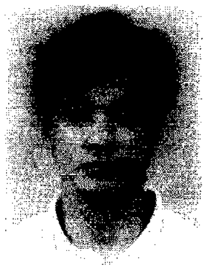
e-mail : mjkim@gamma.kyunghee.ac.kr

1989년 경희대학교 문리대 수학과 졸업(학사)

1992년 경희대학교 전자계산공학과 졸업(석사)

1992년~현재 경희대학교 전자계산공학과 박사과정 재학중

관심분야 : 인공 지능, 음성합성



안 종 일

e-mail : jiahn@gamma.kyunghee.ac.kr

1992년 국민대학교 기계설계학과 졸업(학사)

1994년 경희대학교 대학원 전자계산공학과(공학석사)

1994년~현재 경희대학교 대학원 전자계산공학과 박사과정 재학중

관심분야 : 인공 지능, 신경망 이론 등.

정 태 충

e-mail : tcchung@nms.kyunghee.ac.kr

1980년 서울대학교 전자공학과 졸업(학사)

1982년 한국과학기술원 전산계산학과(공학석사)

1987년 한국과학기술원 전산계산학과(공학박사)

1987년 9월~1988년 3월 KIST시스템 공학센터 선임연구원

1988년 3월~현재 경희대학교 전자계산공학과 교수

관심분야 : 인공 지능, 자연어 처리, 멀티미디어 등