전광 트리 네트워크에서 파장 및 경로설정 문제를 해결하는 알고리즘에 관한 연구

김 순 석[†]·여 상 수[†]·김 성 권[††]

요 약

본 논문은 전광 트리 네트워크상에서 WDM(Wavelength Division Multiplexing)을 이용한 파장 할당 및 경로 설정문제 즉, WRA 문제를 고려한다. 현재 전광 네트워크상에서 요청 노드쌍들을 연결하는 각각의 링크에는 서로 다른 파장을 할당해야 한다는 기술상에 제약조건을 안고 있다. 이것에 기반하여 본 논문에서는 일반트리구조상에서 분할정복방법을 이용하여 실제 모든 경로에 파장을 할당하는 다항시간 알고리즘을 제안한다. 제안한 알고리즘의 분석 결과는 $O(Q \cdot R)$로, 여기서 Q는 일반트리구조상의 모든 노드에서 다른 모든 노드로 가는 요청 노드쌍들의 수이며 R는 일반트리구조상에서 모든 경로에 대해 할당할 수 있는 최대 파장 수이다. 또한, 제안한 알고리즘을 펜티엄II 233MHz에서 C언어로 구현하여 노드 수와 파장 수 그리고 수행시간에 대한 실험 결과를 제시한다.

A Study on Algorithm for the Wavelength and Routing Assignment Problem on All-optical Tree Networks

Soon-Seok Kim[†] · Sang-Su Yeo[†] · Sung-Kwon Kim[††]

ABSTRACT

This paper considers the WRA(Wavelength and Routing Assignment) problem on all-optical tree networks using WDM(Wavelength Division Multiplexing). Each link between a pair of request nodes on all optical networks is assigned different wavelengths because of technical constraint. On the basis of this, we give an polynomial time algorithm to assign wavelengths to the all paths of a arbitrary tree network using divide and conquer method. The time complexity of this algorithm is $O(Q \cdot R)$, in which Q is the request nodes for all-to-all communication in a tree topology and R is the maximum number of wavelength. Also we implemented our algorithm using C in Pentium II 233MHz and analyzed performance results.

1. 서 론

광 기술은 하나의 광 파장만으로 초당 수 기가 바이트나 되는 전송률을 지원하고 있어서 음성, 데이터 그리고 비디오 등과 같은 다중의 채널을 동시에 서비스할 수 있을 뿐만 아니라, 각기 다른 빛의 파장으로 동일한 광섬유 링크를 따라 전파되는, 여러 개의 레이저 빔들을 이용하게 되면 보다 나은 전송률을 제공할 수가 있다. 이러한 기술은 모두가 광 대역폭을 여러 개의 채널로 나누어서, 이 여러 개의 데이터 스트림이 동시에 같은 광섬유 링크내의 다른 채널을 통해 전송되는 파장 분할 다중화(WDM)방식을 이용함으로써 가능하게 되었는데, 현재 WDM을 이용한 네트워크의 핵심 기술로는 다중화 가능한 파장의 수를 늘려 전송 라인의 대역폭을 확장하는 것과 각 노드의 처리 용량을 늘리고 파장 사용의 효율성을 높이기 위한 파장 라우

※ 본 연구 결과는 정보통신부 정보통신연구진흥원의 대학정보통신연구센터 육성 사업의 일환으로 수행되었음.
† 준 회 원 : 중앙대학교 대학원 컴퓨터공학과
†† 종신회원 : 중앙대학교 컴퓨터공학과 교수
논문접수 : 2000년 1월 31일, 심사완료 : 2000년 12월 18일

팅 기술을 들 수 있다. 그러나 동시에 다중화 가능한 파장의 수에는 제한이 따르며 망의 건설 비용을 절감하기 위해서도 최소 개수의 파장만을 사용하여 주어진 통신 요구를 만족시킬 수 있도록 망을 구성하여야 한다. 따라서, 파장 할당 및 파장 경로의 라우팅 즉, WRA(Wavelength and Routing Assignment) 문제는 이러한 최적화된 WDM 네트워크를 구성하는데 있어서 반드시 해결되어야 하는 문제이다[7].

광 네트워크에서 주어진 요청 경로들의 집합을 각각 (a_1, b_1), (a_2, b_2), (a_3, b_3), $\cdots$, (a_k, b_k)라 하면, 경로 P_i에 의해 노드 a_i와 b_i가 연결될 것이 요구되는데, 이때 전자기적인 간섭 현상을 피하기 위해 같은 파장을 갖는 모든 경로들이 한 링크를 공유하지 않도록 각각의 경로 P_i에 파장을 할당할 수 있다. 이러한 관점에서 문제를 보면, 이 문제는 무방향(undirected) 그래프[1] 상의 문제로 생각할 수 있다. 그러나, 근래에는 양방향 링크가 서로 반대 방향의 두 단방향 링크로 구성되어 있다는 관점에서 보고 네트워크를 단순히 무방향 그래프로 생각하는 것이 아니라 방향을 갖는 경로(directed path)로 생각하여 문제를 해결하는 것이 더 바람직하다는 견해가 많아 그 방향으로 연구가 진행되고 있다. 따라서, 이러한 상황에 맞는 새로운 모델을 대칭성 방향 그래프(symmetric directed graph), 내지는 앞서 설명한 방향을 갖는 경로, 줄여서 "dipath"로 간주하여 네트워크의 형태를 표현하고 있다[4, 7].

본 논문에서는 이러한 네트워크의 유형을 트리의 경우로 한정지어 생각하려한다. 그 이유는 사실상 트리 형태의 네트워크가 통신 업계에서 거의 표준으로 자리잡고 있으며[6], 요구되는 노드 쌍들의 경로를 설정하는데 있어서도 트리 구조인 경우에는 그 경로가 유일하기 때문에 문제의 영역을 거의 반으로 줄여줄 수가 있다. 트리의 경우는 크게 이진 트리와 일반 트리의 두 경우로 나뉠 수 있다. 이진 트리의 경우에 있어서는 현재까지 많은 연구가 이루어지고 있으나[2], 일반 트리의 경우에 있어서는 아직 그 연구가 미비한 상태다. 왜냐하면 일반 트리, 특히 그 중에서도 대칭성 방향 트리에서, 이미 네트워크가 주어져 있을 경우에 요청되는 임의 경로들에 최소의 파장수로 파장을 할당하는 문제는 현재까지 몇몇 휴리스틱한 알고리즘들이 개발되고 있긴 하지만, 일반적으로 NP-hard임이 알려져 있기 때문이다[2, 4, 5]. 하지만, 만일 주어진 경로가 임의 경로가 아닌 모든 경로일 경우는 더 이상 NP-hard가 아니다.

대칭성 방향 트리라 함은, 주어진 트리 상의 양방향 링크를 서로 반대 방향의 두 단방향 링크로 간주하고 이를 네트워크 토폴로지로 구성한 것을 말한다. 이때 반드시 주의해야 할 점은 같은 방향에서 하나의 에지를 통과하는 *dipath*는 서로 다른 컬러를 사용해야 한다는 것이다. 이러한 내용을 좀더 정형화하여 기술하면 다음과 같다.

주어진 트리를 T, 트리 T의 *dipath*들의 집합을 S라 하고, 임의의 두 노드를 각각 x, y라 하자. 이때 각 에지는 x로부터 y방향으로 운행한다고 가정한다. 다시 말해, *dipath* $P(x, y)$와 $P(y, x)$는 서로 다르며, 같은 방향에서 하나의 에지를 이용하여 S의 *dipath*들이 서로 다른 컬러를 얻을 수 있도록 주어진 S를 채색하는 것이다. 이러한 문제와 관련하여 [5]에서는 다음과 같은 결과를 정리하고 증명하였다.

[보조정리 1] [5] 주어진 트리를 T라 하고 트리 T의 *dipath*들의 집합을 S라 하자. 이때 만일, 주어진 S를 채색하는 데 사용될 수 있는 최소 컬러 수(혹은 파장 수)를 $c(T)$라 하고 주어진 트리 T의 임의의 한 에지를 동일 방향으로 통과하는 S의 *dipath*들의 최대 수를 $\Pi(T)$라 한다면, $c(T)=\Pi(T)$이다.

즉, 위 [5]에서는 파장 할당 문제를 경로 채색 문제로 간주하여 주어진 문제를 해결하고 있는데, 주어진 네트워크 토폴로지를 일반 트리라 가정하고 트리상의 모든 경로에 대해 파장을 할당한다고 할 경우에, 해당하는 모든 경로에 대해 채색(coloring)할 수 있는 최소 컬러 수는 일반 트리구조상에서 모든 경로에 대해 할당할 수 있는 최대 파장 수와 같음을 증명하고 있다. 즉, 최대 파장 수($c(T)$ 혹은 $\Pi(T)$)의 범위 내에서 원하는 임의의 모든 경로에 파장을 할당할 수 있다는 것을 의미한다.

참고 논문 [5]의 경우에 있어서는 현재 위의 보조정리 1에서 언급한 정리와 이 정리를 증명하는 과정에서 효율적인 다항시간 알고리즘을 부분적으로 제시하고 있다. 그러나, 실제 세부적인 알고리즘이 어떻게 이루어지는지에 대한 세부적인 제시라든가 제시한 알고리즘에 대한 정확성 그리고 수행시간 등에 대한 분석이 미비하다.

따라서, 본 논문에서는 주어진 경로가 임의 경로가 아닌 모든 가능한 경로(all-to-all)에 대해, 앞서 설명한

[5]에서 증명된 이론을 근거로, [5]에서 제시하고 있는 부분적인 알고리즘과는 다른 다항 시간 내에 해결 가능한 실질적인 알고리즘을 제안한다. 또한, 제안한 알고리즘에 대한 정확성을 증명하고 수행시간에 대해 분석한 후, 이를 펜티엄II 233MHz에서 C언어로 실제 구현하여 실험한 결과를 제시하고자 한다.

2. 제안하는 WRA 알고리즘

2.1 알고리즘 개요

본 논문에서 제안하는 알고리즘은 위의 보조정리 1을 근거로 일반 트리 상에서 모든 경로에 파장을 할당하는 알고리즘이다. 그러기 위해선 먼저, 최대로 필요한 파장의 개수($c(T)$ 혹은 $\Pi(T)$)를 알아야 하며 이 두 수를 근거로 파장을 할당해야 한다. 이때, 이 두 값은 모두 동일함으로, 이 둘 중 어느 하나의 값만 알아도 나머지의 값은 쉽게 구할 수가 있다.

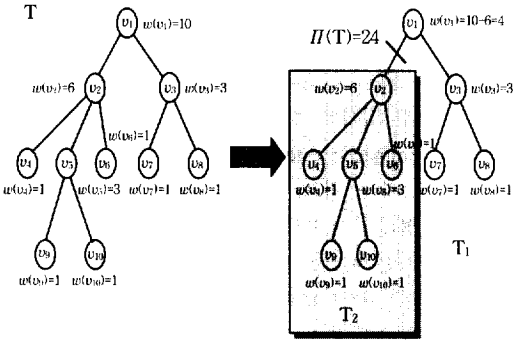
지금부터 우리는 $\Pi(T)$의 값을 알고자 한다. $\Pi(T)$는 주어진 트리 T의 에지들 중, 가장 많은 경로를 포함하는 에지를 찾아 그 경로 수를 구하면 된다. 일반적으로 임의의 어느 한 에지가 얼마나 많을 경로를 갖는가를 알기 위해서는 주어진 트리 T를 해당 에지를 중심으로 양분하여 양분된 각각의 서브트리 내에 있는 각 노드들의 합을 구해 이 둘을 곱하면 된다. 따라서, 이 문제는 주어진 트리 T의 각 노드마다, 이 노드를 근노드로 하는, 자신 노드를 포함한 하위 노드들의 최대 개수 즉, 가중치를 조사함으로써 해결할 수가 있다.

주어진 트리를 T, 트리 T의 임의의 한 노드를 v라 하고 노드 v의 가중치를 $w(v)$라 하자. 이때 $w(v)$는 자신 노드를 포함한 하위 노드들의 개수의 합으로, 양의 정수이다. 만일 트리 T내에 임의의 노드들의 집합을 V라 한다면,

$$w(V) = \sum_{v \in V} w(v) \qquad (1)$$

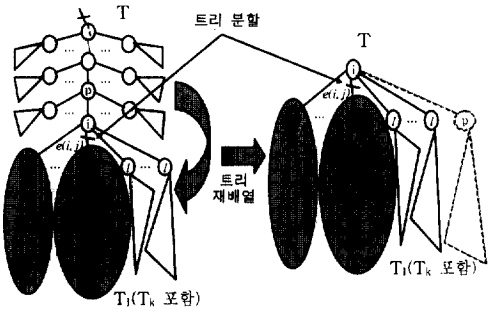
로 표기할 수 있다.

주어진 트리 T의 임의의 어느 한 에지를 e라 하자. 주어진 트리 T로부터 에지 e를 제거하면 각각 두 개의 서브트리 T_1과 T_2로 나뉠 수 있다. 이것은 어느 한 에지 e를 중심으로 주어진 트리 T가 두 개의 서브트리 T_1과 T_2로 양분됨을 의미한다. 에지 e를 중심으로 링크된 두 노드들 중 서브트리 T_1에 속해있는 노드를 v_1이라 하고 서브트리 T_2에 속해있는 노드를 v_2라 하자. 우리가 구하고자 하는 $\Pi(T)$의 값은 주어진 트리 T의 모든 에지 e들을 중심으로 $w(v_1) * w(v_2)$의 값을 조사하여 이 값들 중 최대 값을 구하면 된다. 이때, 주어진 트리 T의 전체 노드 수를 n이라 한다면 $n = w(v_1) + w(v_2)$인 식이 성립한다. 따라서, 실제 최대값은 $w(v_1)$과 $w(v_2)$ 둘 중 하나의 값이 $n/2$의 값에 가장 근사한 정수값이 될 것이다. 만일, 이 값을 $w(v_1)$이라 한다면 나머지 $w(v_2)$의 값은 자동적으로 $n - w(v_1)$이 된다. 이 두 값을 구해 $w(v_1) * w(v_2)$한 값이 바로 $\Pi(T)$ 즉, 주어진 트리 T의 임의의 한 에지를 동일 방향으로 통과하는 *dipath*들의 최대 수이며, $c(T)$ 즉, 주어진 *dipath*들의 집합 S를 채색하는 데 사용될 수 있는 최소 컬러 수(혹은 파장 수)가 된다. 아래 (그림 1)은 주어진 트리 T의 전체 노드 수가 10일 때 각 서브트리 T_1과 T_2들에 대해 필요한 최대 파장수 즉, $\Pi(T)$값을 구한 예이다.



(그림 1) 주어진 트리 T에서의 최초 분할

지금까지 주어진 트리 T에서 최대 경로수를 갖는 에지 e와 이 에지를 중심으로 어떻게 각 서브트리가 양분되는지에 대해 살펴보았다. 그러나, 필요한 에지를 찾아 각 서브트리로 분할하기 이전에 먼저 주어진 트리 T를 재배열하는 과정이 필요하다. 재배열하는 방법은 아래 (그림 2)와 같이 노드 i의 부모노드인 노드 p와 그 상위에 있는 모든 노드들을 노드 i의 자식노드와 그 하위노드들로 주어진 트리를 변형한다. 이렇게 하는 이유는 일반트리의 여러 유형 가운데 사향트리라든가 그밖에 특수한 형태의 위상을 갖는 트리를 주어진 컬러링에 보다 쉽게 하기 위해 일반적인 형태로 변형하기 위해서이다.



(그림 2) 부분 컬러링을 위한 트리 분할 및 재배열

2.2 알고리즘

지금까지 설명한 알고리즘 개요를 토대로 실제 알고리즘을 기술하면 다음과 같다.

단계 1. 주어진 트리 T의 전체 노드들에 대한 가중치 $w(1), w(2), w(3), \cdots, w(v), \cdots w(n)$(이때, v는 트리 T내의 임의노드이며 n은 트리 T의 전체 노드수이다.)을 구하여 최대 경로 수를 갖는 에지 $e(i,j)$를 찾는다.

단계 2. 위의 에지 e를 중심으로 링크된 두 노드 i와 j 중 가중치가 보다 크거나 같은 노드를 근노드로 주어진 트리 T를 재배열한 후, 위의 단계 1을 통해 구해진 특정 에지 $e(i,j)$를 중심으로 서브트리 T_1과 T_2로 각각 양분한다.(만일 $w(i) >= w(j)$이면 노드 i와 j는 각각 서브트리 T_1과 T_2의 근노드가 된다.)

단계 3. 만약 트리 T의 모든 경로에 부여된 컬러가 하나도 없다면 전역 컬러링을 행하고, 그렇지 않으면 서브트리 T_1과 T_2에 대해 각각 부분 컬러링을 행한다.

단계 4. 주어진 서브트리 T_1과 T_2를 각각 T라 놓고, 트리 T가 더이상 양분되지 않을 때까지 각각의 서브트리에 대해 위의 단계 1에서부터 3까지를 반복해서 행한다.

위의 알고리즘에서 기술한 바와 같이 주어진 일반 트리 T의 모든 경로에 컬러를 할당하는 방법은 크게 두 가지로 나뉠 수 있다.

① 전역 컬러링 - 최초로 양분된 서브트리 T_1과 T_2에 대해, 제한된 최대 컬러 수 $\Pi(T)$의 범위 내에서 일정 순서에 의해 파장을 할당한다.

② 부분 컬러링 - 최초가 아닌 양분된 서브트리 T_1과 T_2에 대해, 제한된 최대 컬러 수 $\Pi(T)$의 범위 내에서 다음의 제약 조건을 만족하면서 일정한 규칙에 의해 파장을 할당한다.

[제약조건] 주어진 네트워크 상에서 하나의 에지를 같은 방향으로 공유하는 어떤 두 경로도 동일한 파장으로 할당될 수 없다.

[전역 컬러링 방법]

전역 컬러링 방법은, 주어진 트리 T를 최초 양분한, 서브트리 T_1과 T_2에 대해 다음과 같은 방법으로 파장을 할당한다.

- 서브트리 T_2의 전체 노드에서 서브트리 T_1의 전체 노드로의 컬러 할당
  ① 서브트리 T_2의 근노드로부터 T_1의 전체 노드로 DFS(깊이우선탐색, Depth First Search)순서를 따라 *컬러 1*부터 차례로 *컬러 2, 컬러 3, …, 컬러 $w(T_1)$*(이때, $w(T_1)$은 가중치로 서브트리 T_1의 전체 노드의 개수)을 서브트리 T_1의 전체 노드 개수만큼 할당한다.
  ② DFS의 순서에 따라 서브트리 T_2의 다음 노드를 검색하여 이 노드부터 서브트리 T_1의 전체 노드로, 위의 단계1에서 할당된 다음 컬러부터 차례로 적용해나간다.
  ③ 위의 과정을 서브트리 T_2의 마지막 노드까지 반복하여 $\Pi(T)$의 수만큼 컬러를 할당한다.

- 서브트리 T_1의 전체 노드에서 서브트리 T_2의 전체 노드로의 컬러 할당
  ① 서브트리 T_2의 전체 노드에서 서브트리 T_1의 전체 노드로의 컬러 할당의 역순으로 컬러를 할당한다. 즉, *컬러1*은 *컬러 $\Pi(T)$*로 *컬러 2*는 *컬러 $\Pi(T)$-1*로 *컬러 3*은 *컬러 $\Pi(T)$-2*로 계속해서 할당하여 *컬러 1*이 될 때까지 할당한다.

[부분 컬러링 방법]

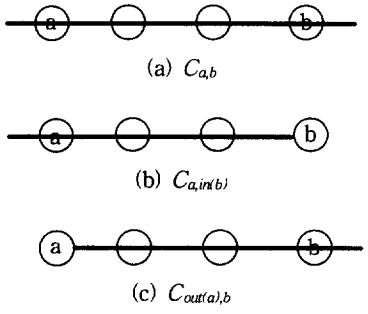
부분 컬러링 방법은 앞서 설명한 바대로 최초의 전역 컬러링이 행해진 이후에 각 서브트리 T_1과 T_2에 대해 행해진다. 따라서, 이때 이용된 각 서브트리 T_1과 T_2를 다시 트리 T라 놓고 트리 T에서 최대 경로 수를 갖는 특정 에지 $e(i,j)$를 중심으로 양분해서 분할된 두 서브트리의 전체 노드들간에 컬러링을 행하게 된다.

지금부터는 최초의 주어진 트리가 아닌 최소한 한번의 분할이 이루어진 서브트리를 T라 하고 트리 T를

양분하여 분할된 각 서브트리를 T_1, T_2라 하기로 한다. 이때, 노드 j는 위의 (그림 2)에서 보는 바와 같이 노드 i의 자식노드로 서브트리 T_2의 근노드이며 노드 i 역시 주어진 트리가 재배열과정을 거치고 난 이후에는 항상 서브트리 T_1의 근노드가 된다.

주어진 트리 T에서 노드 j를 포함하지 않는 노드 i의 자식노드들 중 임의의 한 노드를 노드 k라 하고(이때, 노드 k를 근노드로 하는 서브트리를 T_k라 하자.) 노드 k와 노드 j를 제외한 나머지 모든 자식노드들을 노드 l, 노드 j의 모든 자식노드들을 노드 m이라 하자. 이때, 서브트리 T_k의 모든 노드들은 컬러를 할당하고자하는 대상이 되는 노드들로서 서브트리 T_1에 속한 노드들이다. 그밖에, 노드 i의 부모노드를 노드 p라 할 때, 노드 p는 위의 (그림 2)에서 보는 바와 같이 트리가 재배열된 후에는 노드 i의 자식노드로 된다. 이후에 자세히 설명되겠지만, 여기서 노드 k는 노드 i의 자식노드수가 노드 j를 포함하여 2인 경우에 필요하며 노드 l은 노드 j를 포함하여 3이상인 경우에 사용될 것이다.

만일, 여기서 전역 컬러링 또는 부분 컬러링 방법을 통해 할당된 컬러가 $C_{a,b}$에 저장된다고 가정하자. 아래 (그림 3)에서 보는 바와 같이, $C_{a,b}$는 할당된 컬러들의 집합으로, a와 b는 각각 단일 노드이며, 이것은 노드 a에서 링크들을 따라 중간노드들을 거쳐 노드 b에 도착하기까지 할당된 컬러가 저장됨을 의미한다. 또한, 여기서는 $in(v)$(v는 임의의 노드)와 $out(v)$를 사용하였는데, 만일 $C_{a,in(b)}$라 하면 노드 a를 거쳐가되 노드 b를 거치지 않고 노드 b에 도착한 컬러를 의미하며, $C_{out(a),b}$라 하면 노드 a를 거치지 않고 해당 노드 a에서 출발하여 노드 b를 거쳐가는 컬러를 의미한다. 여기서 주의해야 할 점은 컬러가 할당될 때는 반드시 동일 방향의 에지를 통과하는 같은 컬러가 없어야 한다는 것이다. 따라서, 컬러를 할당할 때는 먼저 위의 제약 조건을 만족하는지를 살펴보아야 한다. 하지만, 본 알고리즘에서는 기본적으로 위의 사실에 따르고 있으며, 각 요청 노드쌍들에 대한 컬러의 할당 순서는 위의 전역 컬러링에서와 마찬가지로 DFS 순서에 따른다고 가정한다.



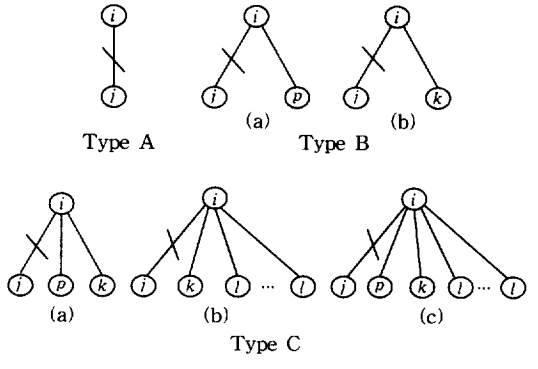
(그림 3) 컬러 $C_{a,b}$, $C_{a,in(b)}$, $C_{out(a),b}$

부분 컬러링 방법은 최초 트리 분할이후부터, 트리 T의 각 요청 노드쌍들에 대해 일정한 제약조건을 만족하면서 채색이 이루어진다. 여기서는 기본적으로 주어진 트리의 유형을 세 가지로 나눠 요청되는 각 노드쌍들에 대해 컬러를 할당하였다. 즉, 주어진 트리가 위에서 말한 특정 에지 $e(i,j)$를 중심으로 양분된다고 할 때, 노드 i의 자식노드 수에 따라 트리의 유형을 아래 (그림 4)와 같이 세 가지로 구분된다.

Type A. 노드 j를 포함한 노드 i의 자식노드 수가 1일 때

Type B. 노드 j를 포함한 노드 i의 자식노드 수가 2일 때

Type C. 노드 j를 포함한 노드 i의 자식노드 수가 3보다 크거나 같을 때



(그림 4) 부분 컬러링 방법에서 트리의 세 유형

또한, 위의 세 가지 유형에 대해 주어진 트리가 재배열되기 이전에 노드 i의 부모노드인 노드 p의 유무에 따라 다시 세부 유형들로 나뉘어 지며, 이 유형들은 우선 순위에 따라 순차적으로 할당해야 할 개수만큼 컬러가 부여된다. 이때, 할당해야할 개수는 $w(i)*w(j)$으로, $w(i)$는 서브트리 T_1의 근노드로 T_1의 전체 노드 개수이고 $w(j)$는 서브트리 T_2의 근노드로 T_2의 전체 노드의 개수이다. 또한, 할당해야할 컬러는 위의 정리 1에 의해 이전의 전역 컬러링에서 행한 *컬러 1*부터 *컬러 2, 컬러 3, … , 컬러 $\Pi(T)$*의 범위 내에서 위

의 제약조건을 만족하면서 행해진다.

- Type A
  - 서브트리 T_2의 전체 노드에서 노드 i로의 컬러링
    $C_{i,in(j)}$, $C_{j,m}$순으로 저장된 컬러들을 검색해 나가되 할당하고자하는 각 경로상에 동일 방향의 컬러가 이미 존재할 경우는 제외하고 컬러를 할당한다.

- Type B

  (a)의 경우
  - 서브트리 T_2의 전체 노드에서 노드 i 그리고 노드 p로의 컬러링
    $C_{p,in(i)}$ 또는 $C_{out(i),p}$ 그리고 $C_{i,in(j)}$, $C_{j,m}$순으로 저장된 컬러들을 검색해 나가되 할당하고자하는 각 경로상에 동일 방향의 컬러가 이미 존재할 경우는 제외하고 컬러를 할당한다. 만일, 노드 p의 하위 노드들이 존재할 경우는 그 하위 노드를 다시 노드 p라 두고 DFS순으로 하위의 전체 노드들에 대해 같은 방법을 적용한다.

  (b)의 경우
  - 서브트리 T_2의 전체 노드에서 노드 i 그리고 서브트리 T_k의 전체노드로의 컬러링
    만일, 서브트리 T_2의 전체 노드에서 노드 i로의 컬러링인 경우는 $C_{i,k}$를 먼저 검색하고 그렇지 않은 경우는 $C_{k,i}$, $C_{i,in(j)}$, $C_{j,m}$순으로 저장된 컬러들을 검색해 나가되 할당하고자하는 각 경로상에 동일 방향의 컬러가 이미 존재할 경우는 제외하고 컬러를 할당한다.

- Type C

  (a)의 경우
  - 서브트리 T_2의 전체 노드에서 노드 i와 노드 p 그리고 서브트리 T_k의 전체노드로의 컬러링
    만일, 서브트리 T_2의 전체 노드에서 노드 i와 노드 p로의 컬러링인 경우는 먼저 $C_{i,k}$ 그리고 $C_{p,in(i)}$ 또는 $C_{out(i),p}$에 저장된 컬러들을 먼저 검색해 나가고 그렇지 않은 경우는 $C_{k,i}$, $C_{i,in(j)}$, $C_{j,m}$순으로 저장된 컬러들을 검색해 나가되 할당하고자하는 각 경로상에 동일 방향의 컬러가 이미 존재할 경우는 제외하고 컬러를 할당한다. 만일, 노드 p의 하위 노드들이 존재할 경우는 그 하위 노드를 다시 노드 p라 두고 DFS 순으로 하위의 전체 노드들에 대해 같은 방법을 적용한다.

  (b)의 경우
  - 서브트리 T_2의 전체 노드에서 노드 i 그리고 서브트리 T_k의 전체노드로의 컬러링
    만일, 서브트리 T_2의 전체 노드에서 노드 i로의 컬러링인 경우는 $C_{i,k}$ 혹은 $C_{i,l}$중 저장된 컬러가 보다 많은 쪽부터 먼저 검색해 나가고 그렇지 않은 경우는 $C_{l,i}$, $C_{i,l}$, $C_{k,i}$, $C_{i,in(j)}$, $C_{j,m}$순으로 저장된 컬러들을 검색해 나가되 할당하고자하는 각 경로상에 동일 방향의 컬러가 이미 존재할 경우는 제외하고 컬러를 할당한다.

  (c)의 경우
  - 서브트리 T_2의 전체 노드에서 노드 i와 노드 p 그리고 서브트리 T_k의 전체노드로의 컬러링
    만일, 서브트리 T_2의 전체 노드에서 노드 i와 노드 p로의 컬러링인 경우는 $C_{i,k}$ 또는 $C_{i,l}$중 저장된 컬러가 보다 많은 쪽부터 시작해서 $C_{p,in(i)}$ 또는 $C_{out(i),p}$순으로 먼저 검색해 나가고 그렇지 않은 경우는 $C_{l,i}$, $C_{i,l}$, $C_{k,i}$, $C_{i,in(j)}$, $C_{j,m}$순으로 저장된 컬러들을 검색해 나가되 할당하고자하는 각 경로상에 동일 방향의 컬러가 이미 존재할 경우는 제외하고 컬러를 할당한다. 만일, 노드 p의 하위 노드들이 존재할 경우는 그 하위 노드를 다시 노드 p라 두고 DFS순으로 하위의 전체 노드들에 대해 같은 방법을 적용한다.

위 Type C의 (b)와 (c)의 경우에서 서브트리 T_k의 전체 노드들에 대한 한번의 컬러링이 이루어지고 난 다음에는 계속해서 다른 노드 l들(혹은 노드 l)을 순서대로 한 노드씩 노드 k라 두고 주어진 서브트리 T_l의 다른 모든 노드들에 대해 같은 방법을 적용한다. 이때, 이전에 존재하던 노드 k는 노드 l로 바뀐다. 따라서, 할당된 컬러의 개수도 늘어나 전부 $w(i)*w(j)$개가 될 것이다.

그밖에, 서브트리 T_2의 전체 노드에서 노드 i로의 컬러링인 경우에 있어서 $C_{i,k}$ 혹은 $C_{i,l}$중 할당된 컬러가 보다 많은 쪽부터 먼저 할당해 나가는데 그 이유는 다음과 같다. 예를 들어 $C_{i,l}$에 저장되어 있는 컬러가 $C_{i,k}$에 저장된 컬러보다 많다고 하자. 이 경우 위의 방법에 의해 $C_{i,l}$이 먼저 할당된다. 이때, 현 컬러링이 끝나 현 노드 k가 l로 바뀌고 이전 노드 l이 k로 바뀌어 서브트리 T_2의 전체 노드에서 서브트리 T_k의 전체노드로 컬러링을 하고자 한다. 이런 경우에 아마도 이전에 이

용되지 않았던 $C_{i,k}$에 대한 할당 기회가 많아질 것이다. 그 반대로 $C_{i,k}$가 먼저 할당될 경우도 마찬가지이다. 즉, 다시 말해 이용하고자 하는 컬러가 많을수록 앞으로 할당할 컬러링에도 보다 많은 기회를 부여할 수 있기 때문이다.

지금까지 설명한 부분 컬러링 방법은 양분된 서브트리들 중 T_2의 전체 노드에서 T_1의 전체노드로의 컬러링이었다. 그 반대 방향에 대한 컬러링 또한 위의 방법과 동일하게 적용되므로 본 논문에서는 생략하였다.

2.3 알고리즘 분석

본 논문에서 제안하고 있는 알고리즘의 시간 복잡도에 대해 살펴보자. 주어진 트리를 T, T 내에 있는 전체 노드의 개수를 n, 주어진 트리 T를 컬러링하는데 필요한 최대 컬러수($\Pi(T)$ 혹은 최대 파장수)를 R이라 하자. 이때, $n-1 \le R \le n^2/4$이다. 우선 단계 1의 경우 주어진 트리에 대해 가중치를 계산하는 문제는 해당 트리의 각 노드를 DFS순으로 방문하면 되므로 시간 복잡도는 $O(n)$이다. 단계 2의 경우 역시 주어진 트리 내에 있는 각 에지를 한번씩 방문하면 되므로 시간 복잡도는 $O(n)$이다. 실질적인 파장이 할당되는 단계 3에서, 전역 컬러링의 경우는 최초 한번만 수행되며, 최대 컬러수 R만큼 할당하면 되므로 시간 복잡도는 $O(R)$이다. 이에 반해 부분 컬러링의 경우에 시간 복잡도는 다음과 같이 계산된다.

[정리 1] 부분 컬러링에서 할당해야할 요청들 즉, 요청 노드쌍들의 개수를 Q라 하면, Q는 다음과 같이 나타낼 수 있다.

$$Q \le 2(2^{2k-3}+2^{2k-4}+2^{2k-5}+\cdots+2^{k-1})$$

이때, $3 \le n \le 2^k$이며 k는 2보다 큰 정수이다.

[증명]

요청 노드쌍들의 개수가 최대가 될 경우는 주어진 트리 T를 중심으로 양분이 일어날 때, 양분된 서브트리 각각의 개수가 정확히 반으로 분할될 때이다. 따라서, 노드의 개수 n에 대해 각각 $n/2$, $n/2^2$, $n/2^2$, $\cdots$, $n/2^k$, $\cdots$, n/n으로 서브트리의 개수가 분할된다. 이때, k는 2보다 큰 정수가 되는데 그 이유는 부분 컬러링이 최초 분할 이후 즉, 두 번째 컬러링부터이기 때문에 실질적인 부분 컬러링은 서브트리의 개수가 $n/2^2$ 즉, k가 2인 경우부터가 된다. 또한, n은 아무리 분할된다 하더라도 최대 2^k보다 클 수가 없다.

이러한 서브트리의 개수를 근거로 부분 컬러링에서 각 요청 노드쌍들의 개수를 구해보자.

주어진 트리 T를 두 번째 분할하게 되면 4개의 서브트리들로 나뉘어지며 이들 중 각각 두 개의 서브트리들간에 최초의 부분 컬러링이 이루어진다. 이때, 각 서브트리의 개수는 앞서 설명한 바와 같이 각각 최대 $n/4$으로 이것은 다시 $\frac{2^k}{2^2}$으로 쓸 수 있다. 따라서, 두 개의 서브트리간에 $\frac{2^k}{2^2}*\frac{2^k}{2^2}$개의 요청 노드쌍들에 대한 컬러링이 필요하고, 나머지 두 개의 서브트리에도 $\frac{2^k}{2^2}*\frac{2^k}{2^2}$개의 컬러링이 요구되며 이를 양방향으로 계산하면 $2\left\{2^1(\frac{2^k}{2^2}*\frac{2^k}{2^2})\right\}$개의 요청 노드쌍들이 필요하다. 또한, 분할이 계속될 때마다 할당해야할 요청들의 개수는 각각 $2\left\{2*2(\frac{2^k}{2^3}*\frac{2^k}{2^3})\right\}$, $2\left\{2*2*2(\frac{2^k}{2^4}*\frac{2^k}{2^4})\right\}$, $2\left\{2*2*2(\frac{2^k}{2^4}*\frac{2^k}{2^4})\right\}$, $\cdots$, $2\left\{2^{k-1}(\frac{2^k}{2^k}*\frac{2^k}{2^k})\right\}$으로 줄어들게 된다. 그러므로, 이를 식으로 나타내면,

$$Q \le 2\left\{ 2^1(\frac{2^k}{2^2}*\frac{2^k}{2^2})+2^2(\frac{2^k}{2^3}*\frac{2^k}{2^3})+\cdots+ 2^{k-1}(\frac{2^k}{2^k}*\frac{2^k}{2^k})\right\}$$

$Q \le 2(2^{2k-3}+2^{2k-4}+2^{2k-5}+\cdots+2^{k-1})$이다. □

위 [정리 1]에서 살펴본 요청 노드쌍 Q에 대해 부분 컬러링에서는 2.2절의 알고리즘에서 제시한 바와 같이 한번의 컬러링이 있을 때마다 필요한 최대 파장수 즉, R의 범위 내에서 각각에 대한 검색이 이루어짐으로 $O(Q \cdot R)$만큼의 시간 복잡도가 요구된다. 또한, 이 복잡도는 본 논문에서 제시하고 있는 알고리즘 전체의 시간 복잡도 이기도 하다. 이를 주어진 트리 T의 노드의 개수 n에 대해 계산하면 아래 [정리 2]와 같이 나타낼 수 있다.

[정리 2] 주어진 트리를 T라 하고 트리의 모든 노드의 개수를 n이라 하자. 이때, 본 논문에서 제시하고 있는 알고리즘의 시간 복잡도는 $O(n^4)$이다.

[증명]

앞서 [정리 1]에서 언급한 바대로 최악의 경우에 시

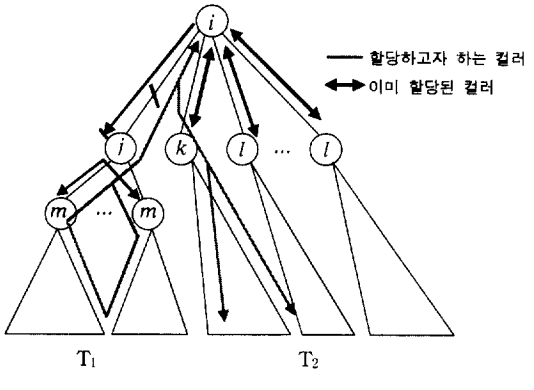
간 복잡도는 Q가 $2(2^{2k-3}+2^{2k-4}+2^{2k-5}+\cdots+2^{k-1})$ 이고, R이 $\frac{n^2}{4}$ 일 경우이다. 따라서,

$$O(Q*R)=O(\{2(2^{2k-3}+2^{2k-4}+2^{2k-5}+\cdots+2^{k-1})\}*\frac{n^2}{4})$$
$$=O(2^{2k}*n^2)$$
$$=O(2^{2}\log n*n^2)$$
$$=O(n^2*n^2)$$
$$=O(n^4). \square$$

2.4 알고리즘의 정확성

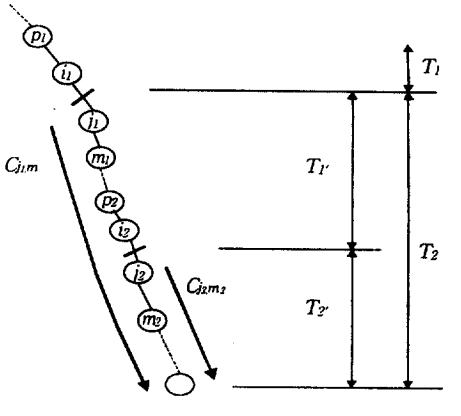
[정리 3] 본 논문에서 제시하고 있는 알고리즘은 앞 절에서 정의한 R 즉, 최대 파장수(또는 컬러수)의 범위 내에서 주어진 트리 T의 모든 경로에 파장을 할당한다.

[증명] 본 논문의 2.2절에서 언급한 바대로 전역 컬러링의 경우는 먼저, 주어진 트리 T를 각각 두 개의 서브트리로 나누어 컬러 1부터 최대 컬러수 R만큼 파장을 할당하고 있으며 부분 컬러링에서도 역시 이전의 전역 컬러링에서 이미 할당한 컬러들을 가지고서 이후 분할된 모든 서브트리들간의 컬러링에 이용하고 있다. 따라서, 본 논문에서 제시하고 있는 알고리즘은 최대 파장수 R의 범위내에서 모든 경로에 파장을 할당한다. □



(그림 5) 서브트리 T_2의 전체노드에서 서브트리 T_1의 전체 노드로의 컬러 할당의 예

위 2.2절에서 설명한 부분 컬러링 방법의 유형 중 Type C의 (b)경우인 서브트리 T_2의 전체 노드에서 서브트리 T_1의 전체 노드로의 컬러링을 고려해 보자. 이 경우 위의 (그림 5)와 같이, 만일 서브트리 T_2의 전체 노드에서 노드 i로의 컬러링인 경우는 $C_{i,k}$ 또는 $C_{i,l}$중 저장된 컬러가 보다 많은 쪽부터 먼저 검색해 나가고 그렇지 않은 경우는 $C_{i,i}$, $C_{i,l}$, $C_{k,i}$, $C_{i,in(j)}$, $C_{j,m}$순으로 저장된 컬러들을 검색해 나가되 할당하고자 하는 각 경로상에 동일 방향의 컬러가 이미 존재할 경우는 제외하고 컬러를 할당해 나간다. 이때, 마지막으로 이용하려는 $C_{j,m}$ 즉, 노드 j에서 노드 m으로 가는 경로에 할당된 컬러들은 서브트리 T_1이 다시 분할되면서 서브트리 T_1의 하위에 있는 서브트리들(각각 T_1'와 T_2'라 하자.)간의 컬러링시에, 이용하려는 컬러가 이미 동일 경로에 할당되어있는 상황이 발생한다. 만일, 이러한 경우가 최악으로 발생한다고 가정하면, 아래 (그림 6)과 같은 유형이 될 것이다. 왜냐하면 노드 i_1과 i_2의 자식 노드가 각각 j_1과 j_2만 존재할 경우 $C_{j1,m1}$과 $C_{j2,m2}$에 할당된 컬러들을 이용할 수 있는 가능성이 제일 높기 때문이다. 이러한 상황에서 하위의 두 서브트리 T_1'와 T_2'간에 부분 컬러링을 할 경우를 고려해 보자. 최악의 경우를 가정하면 $C_{j2,m2}$에 할당된 컬러들을 우리가 이용해야하는데 상위의 두 서브트리간(T_1과 T_2간)의 컬러링시에 이미 필요한 컬러수만큼 이용이 되었을 것이다. 따라서, 이때 남은 컬러들을 가지고서 현 요청 노드쌍들을 모두 컬러링해야한다. 이러한 조건을 만족하기 위해서는 $C_{j2,m2}$에 저장된 컬러수를 S라 하고 상위의 두 서브트리 T_1과 T_2간의 컬러링시에 $C_{j1,m1}$에 저장된 컬러들 중 실제 컬러링에 이용된 컬러수를 U라 하며 $C_{j2,m2}$에 저장된 컬러들 중 하위의 두 서브트리 T_1'와 T_2'간의 컬러링시에 실제 필요한 컬러수를 L이라 할 때, $S-U>L$인 관계식이 성립해야한다.



(그림 6) 중복된 컬러링의 예

먼저, 서브트리 T_2의 전체 노드에서 서브트리 T_1의 전체 노드로의 컬러링을 고려하면, 임의 요청 노드쌍들

의 개수는 앞서 정리 1에서 언급한 바대로 $\frac{n}{2^k}*\frac{n}{2^k}$, $2 \le k, 3 \le n \le 2^k$이며 서브트리 T_2'의 전체 노드에서 서브트리 T_1'의 전체 노드로의 컬러링을 고려하면 $\frac{n}{2^{k+1}}*\frac{n}{2^{k+1}}$, $2 \le k, 3 \le n \le 2^k$이다. 또한, 이 경우는 부분 컬러링 중에서도 Type B의 (a)에 해당되므로 각각 $C_{p1,in(i1)}$ 또는 $C_{out(i1),p1}$ 그리고 $C_{i1,in(j1)}$과 $C_{j1,m1}$에 저장된 컬러와 $C_{p2,in(i2)}$ 또는 $C_{out(i2),p2}$ 그리고 $C_{i2,in(j2)}$와 $C_{j2,m2}$에 저장된 컬러들의 순으로 각 요청 노드쌍들의 개수만큼 할당될 것이다. 이때, $C_{p1,in(i1)}$과 $C_{out(i1),p1}$ 그리고 $C_{i1,in(j1)}$에 저장된 컬러들의 개수를 α라 하면, α는 $3(\frac{n}{2}+\frac{n}{2^2}+\frac{n}{2^3}+\cdots+\frac{n}{2^{k-1}})=3n(1-2^{1-k})$이며, $C_{p2,in(i2)}$와 $C_{out(i2),p2}$ 그리고 $C_{i2,in(j2)}$에 저장된 컬러들의 개수는 $\alpha+3(\frac{n}{2^k})$이다. 왜냐하면 $C_{p2,in(i2)}$와 $C_{out(i2),p2}$ 그리고 $C_{i2,in(j2)}$에 저장된 컬러들의 경우는 이미 이전에 서브트리 T_1과 T_2로의 분할시에 서브트리 T_1의 전체노드들($\frac{n}{2^k}$)에서 노드 i_2와 j_2로 혹은 노드 i_2에서 서브트리 T_1의 전체 노드들로 오고가는 컬러들이 각각 한번 더 할당되었기 때문이다.

그밖에, $C_{j1,m1}$에 저장되어 있는 컬러의 개수를 β라 하면, β는 $(\frac{n}{2^2}-1)\frac{n}{2}+(\frac{n}{2^3}-1)\frac{n}{2^2}+\cdots+(\frac{n}{2^k}-1)\frac{n}{2^{k-1}}$이며 이를 간략히 나타내면, $\frac{1}{6}(1-2^{-2k})n^2-(1-2^{-k})n$이고, $C_{j2,m2}$에 저장된 컬러의 개수 S는 $\beta+(\frac{n}{2^{k+1}}-1)\frac{n}{2^k}$이다. 위 값들을 모두 정리하면 다음과 같다.

- 서브트리 T_2의 전체 노드에서 서브트리 T_1의 전체 노드로의 컬러링
  - ① 할당해야할 컬러수, $Y=\frac{n}{2^k}*\frac{n}{2^k}$, $2 \le k, n \ge 2^k$
  - ② 컬러링시에 이용가능한 컬러수 = $(C_{p1,in(i1)} + C_{out(i1),p1} + C_{i1,in(j1)}) + C_{j1,m1} = \alpha+\beta$
  - ③ $C_{j1,m1}$에 저장된 컬러들 중 실제 컬러링에 이용된 컬러수, $U = Y-(C_{p1,in(i1)} + C_{out(i1),p1} + C_{i1,in(j1)})$
    $= (\frac{n}{2^k}*\frac{n}{2^k})-\alpha$

- 서브트리 T_2'의 전체 노드에서 서브트리 T_1'의 전체 노드로의 컬러링
  - ① 할당해야할 컬러수, $Y' = \frac{n}{2^{k+1}}*\frac{n}{2^{k+1}}$, $2 \le k$, $n \ge 2^k$
  - ② 컬러링시에 이용가능한 컬러수 = $C_{p2,in(i2)} + C_{out(i2),p2} + C_{i2,in(j2)} + C_{j2,m2}$
    $= \left\{\alpha+3(\frac{n}{2^k})\right\}+\left\{\beta+(\frac{n}{2^{k+1}}-1)\frac{n}{2^k}\right\}$
  - ③ $C_{j2,m2}$에 저장된 컬러들 중 실제 컬러링에 이용된 컬러수, $L = Y'-(C_{p2,in(i2)} + C_{out(i2),p2} + C_{i2,in(j2)})$
    $= (\frac{n}{2^{k+1}}*\frac{n}{2^{k+1}})-\left\{\alpha+3(\frac{n}{2^k})\right\}$
  - ④ $C_{j2,m2}$에 저장된 컬러수, $S = \beta+(\frac{n}{2^{k+1}}-1)\frac{n}{2^k}$

따라서, $S-U>L$

$$\Leftrightarrow \left\{ \beta+(\frac{n}{2^{k+1}}-1)\frac{n}{2^k}\right\}-\left\{(\frac{n}{2^k}*\frac{n}{2^k})-\alpha\right\}> \left[(\frac{n}{2^{k+1}}*\frac{n}{2^{k+1}})-\left\{\alpha+3(\frac{n}{2^k})\right\}\right]$$

$$\Leftrightarrow \left\{ (\frac{2^k n^2-2^{k+1}n}{2^{2k+1}})-\frac{n^2}{2^{2k}}+\beta+\alpha\right\}> \left\{(\frac{n^2-3*2^2 n}{2^{2k+2}})-\alpha\right\}$$

$$\Leftrightarrow \frac{(2^{k+1}-5)n^2+(3*2^2-2^{k+2})n}{2^{2k+2}}+\beta+2\alpha>0$$

이때, $\alpha=3n(1-2^{1-k})$이고, $\beta=\frac{1}{6}(1-2^{-2k})n^2-(1-2^{-k})n$이므로,

준식은

$$\frac{(\frac{1}{3}2^{2k+1}+2^{k+1}-\frac{17}{3})n^2+(5*2^{2k+2}+12(1-2^{k+2}))n}{2^{2k+2}}>0$$

이때, $2 \le k$이고 n은 항상 양의 정수이므로 k=2일 때, $13n^2+140n$으로 위의 식은 항상 성립한다. 따라서, 현 컬러링에서 이용할 수 있는 컬러들 중 일부의 컬러들이 이전 컬러링에 이용되었다 할지라도 남은 컬러들을 가지고서 주어진 모든 요청 노드쌍들을 충분히 컬러링 할 수 있다.

[정리 4] 주어진 트리를 T라 할 때, 본 논문에서 제안한 알고리즘은 트리 T상에서 임의의 한 에지를 같은 방향으로 공유하는 어떤 두 경로도 동일한 파장으로 할당될 수 없다는 제약조건을 만족한다.

[증명] 본 논문에서 제시하고 있는 알고리즘 가운데, 먼저 전역 컬러링의 경우는 주어진 서브트리 T1과 T2를 중심으로 컬러 1부터 컬러 R까지 DFS순으로 순차적으로 할당해나감으로 동일 방향의 경로에 할당되는 같은 컬러는 존재하지 않는다, 두 번째로 부분 컬러링의 경우는 앞서 2.2절의 부분 컬러링 방법에서 보는 바와 같이 일종의 욕심쟁이(greedy) 알고리즘으로 주어진 임의 노드쌍들간의 컬러링시에 반대 방향에서 이용할 수 있는 모든 컬러들을 할당하게끔 되어있다. 그러나, 만일 할당하려는 컬러가 이미 동일 방향의 경로에 할당되어 있는 경우는 이를 검색하여 다른 컬러들을 할당하게 되어 있으며, 이때 남아있는 다른 컬러들을 가지고서, 앞서 언급한 바대로, 충분히 컬러링 할 수 있다. □

3. 실험 결과

본 절에서는 제안한 알고리즘의 구현을 위해 펜티엄 IIPC 233MHz의 환경에서 C언어를 이용, 다양한 실험을 통해 만족할만한 결과를 도출할 수 있었다. 본 논문에서 수행한 실험결과는 크게 다음과 같은 세 가지 기준으로 나뉘어 볼 수 있으며,

① 알고리즘 수행시간(Routing Time)
② 이용된 파장 수(컬러 수), λ
③ 이용된 노드 수

또한, 위의 세 가지 기준에 근거하여 아래와 같은 세 가지 경우에 대해 실험하였다.

① 각기 다른 임의의 노드 수와 임의의 파장 수를 입력한 경우
② 동일 노드 수(50개)에 대해 각기 다른 임의의 파장 수를 입력했을 경우
③ 동일 파장 수(300개)에 대해 각기 다른 임의의 노드 수를 입력했을 경우

본 실험에서 노드 수와 파장수를 각각 최대 50개와 300개로 둔 이유는 일반적으로 광 네트워크에서는 스위칭이라든가 그밖에 전송장치에 대한 비용과 성능이 처리될 수 있는 파장의 수에 매우 의존하고 있어, 한 섬유당 이용할 수 있는 파장의 개수가 상당히 제한되고 있기 때문이다. 기 발표된 논문 [8]에 의하면, 한 섬유당 최대 200개의 파장을 전송시킬 수가 있다고 한다. 이에 반해 현재 업계에서 생산, 처리하고 있는 상업용 WDM 멀티플렉서의 경우, 섬유당 파장 수는 최소 4개(Pirelli사)에서 많게는 32개(IBM사) 정도로 그 수가 매우 제한적이다.

아래 <표 1>에 명시된 노드 수는 주어진 트리의 전체 노드 수를 의미하며, 실험의 객관성을 높이기 위해 트리의 디그리라든가 깊이 등의 토폴로지를 랜덤하게 구성하여 본 실험에 적용하였다. 또한, 이 트리는 본 논문에서 제안한 알고리즘에 따라 최초에 각각 서브트리 T_1과 T_2로 양분되면서 필요한 최소 파장 수가 결정된다. 따라서, 주어진 트리를 컬러링하는데 필요한 최소 파장 수 즉, 최대 경로 수는 서브트리 T_1의 노드 수와 서브트리 T_2의 노드 수를 곱한 값이 된다. 이것은 이후에 나오는 <표 2>와 <표 3>에도 동일하게 적용될 수 있다. 한 예로 아래 <표 1>에서 노드 수가 20개인 경우, 서브트리 T_1과 T_2의 개수는 각각 4와 5가 되며 이 두 수를 곱한 값 즉, 20이 주어진 트리를 컬러링하는데 필요한 최소 파장 수가 된다. 이 때, 주어

〈표 1〉 각기 다른 임의 노드 수와 임의 파장 수를 입력한 경우의 수행시간

노드수	서브트리T_1 노드수	서브트리T_2 노드수	요구되는 최소 파장수	수행시간(초)
2	1	1	1	0
5	3	2	6	0
10	5	5	25	0
20	16	6	96	0.02
30	20	10	200	0.19
35	21	14	294	0.47
40	27	13	351	1.482
45	27	18	486	2.373
50	34	16	544	3.124

〈표 2〉 동일 노드 수(50)에 대해 각기 다른 임의 파장수를 입력했을 경우의 수행시간

노드수	서브트리T_1 노드수	서브트리T_2 노드수	요구되는 최소 파장수	수행시간(초)
50	45	5	225	0.361
	40	10	400	0.371
	35	15	525	0.381
	30	20	600	0.441
	25	25	625	0.861

〈표 3〉 동일 파장 수(300)에 대해 각기 다른 임의 노드수를 입력했을 경우의 수행시간

노드수	서브트리T_1 노드수	서브트리T_2 노드수	요구되는 최소 파장수	수행시간(초)
37	25	12	300	0.08
40	30	10		0.11
56	50	6		0.57
65	60	5		1.251
79	75	4		3.806

진 트리의 전체 노드 20개를 컬러링하는데 수행되는 시간은 0.02초임을 실험을 통해 알 수 있다.

지금까지의 실험 결과들을 조합해 볼 때, 위의 세 가지 경우 모두가 주어진 노드의 개수와 요구되는 파장 수가 많을수록 수행 시간이 이에 비례하여 길어지는 것을 관찰할 수 있었다. 이것은 바꿔 말해 전광 트리 네트워크상에 모든 경로를 컬러링하는 데 있어서, 해당 노드 수와 필요한 최소 파장 수가 전체 수행시간에 커다란 영향을 미친다는 것을 알 수 있다.

4. 결론 및 향후 연구 방향

광 네트워크 기술은 향후 다가올 초고속 정보화 사회를 생각해 볼 때, 가장 중요한 요소 기술 중의 하나로 떠오르고 있다. 특히, 소프트웨어적인 측면에서 볼 때, 전광 네트워크 분야에서 해결해야할 중요한 기술 중의 하나가 바로 주어진 경로를 설정하는데 있어 최소 수를 이용하여 효율적으로 파장을 할당하는 문제이다. 현재 많은 네트워크의 위상 구조가 복잡한 구조를 갖는 것이 아니라 트리와 같은 단순한 위상 구조를 갖는 경우가 많다. 따라서, 본 논문에서는 네트워크의 위상 구조를 트리의 경우로 한정지어, 이러한 일반 트리 상에서 주어진 모든 경로에 대해 파장을 할당하는데 있어 최소 수의 파장만을 이용하여 다항 시간 내에 해결할 수 있는 알고리즘을 제안, 이를 소프트웨어로 구현하였다. 실험 결과 노드수와 파장수가 전체 수행시간에 큰 영향을 미친다는 사실이 밝혀졌다.

제안한 알고리즘의 시간 복잡도는 $O(Q \cdot R)$로 계산되었는데, 논문 [5]에서 부분적으로나마 언급되고 있는 알고리즘이 효율적이라고는 하나 실제 수행시간이라든가 정확성에 대한 분석이 미비하여 본 논문에서 제안하고 있는 알고리즘과의 효율성에 대한 비교 분석이 아쉽다. 향후 이 부분에 대한 비교 분석이 이루어 질 경우 효율성에 대한 언급이 가능할 것으로 기대된다.

그밖에, 만일 요청된 트리에서 일부 링크에 장애가 발생한다든지 아니면, 이미 모든 경로에 파장이 할당되어 있는데도 불구하고 추가적인 연결 요청이 발생한다든지 할 경우에 어떠한 방식으로 파장을 보다 효율적으로 할당 할 것인가가 앞으로 수행해야 할 연구 과제이다.

참 고 문 헌

[1] A. Aggarwal, A. Bar-Noy, D. Coppersmith, R. Ramaswami, B. Schieber, and M. Sudan, "Efficient Routing and Scheduling Algorithms for Optical Networks," in Proceedings of the 5th Annual ACM-SIAM Symposium on Discrete Algorithms(SODA '94), pp.412-423, 1994.

[2] D. Banerjee, and B. Mukherjee, "A Practical Approach for Routing and Wavelength Assignment in Large Wavelength Assignment in Large Wavelength-Routed Optical Networks," IEEE Journal on Selected Areas in Communications, Vol.14, No.5, June 1996.

[3] I. Clamtac, A. Ganz, and G. Karmi, "Purely Optical Networks for Terabit Communication," Proc. IEEE INFOCOM'89, pp.887-896, 1989.

[4] J-C. Bermond, L. Gargano, S. Perennes, A. Rescigno, and U. Vaccaro, "Efficient Collective Communications in Optical Networks," Proc. 23nd ICALP'96, Paderborn, Germany, 1996.

[5] L. Gargano, P. Hell, and S. Perennes, "Colouring Paths in Directed Symmetric Trees with Applications to WDM Routing," Proc. 24nd ICALP'97, 1997.

[6] M. Mihail, C. Kaklamanis, and S. Rao, "Efficient Access to Optical Bandwidth," Proceedings of 36th Annual IEEE Symposium on Foundations of Computer Science(FO-CS'95), pp.548-557, 1995.

[7] P. E. Green. Fiber-Optic Communication Networks, Prentice-Hall, pp.321-339, 1992.

[8] P. E. Green, "Optical Networking Update," IEEE Journal on Selected Areas in Communications, Vol. 14, No.5, pp.764-779, June 1996.

[9] R. Ramaswami, "Multi-wavelength Lightwave Networks for Computer Communication," IEEE Communication Magazine, Vol.31, pp.78-88, Feb. 1993.

[10] R. J. Vetter and D. H. C. Du, "Distributed Computing with High-Speed Optical Networks," IEEE Computer, Vol.26, pp.8-18, 1993.

[11] T. Erlebach and K. Jansen, "Scheduling of Virtual Connections in Fast Networks," Proc. of 4th Workshop on Parallel Systems and Algorithms PASA'96, pp.13-32, 1996.

김 순 석

e-mail : sskim@alg.cse.cau.ac.kr

1997년 진주대학교 컴퓨터공학과 (학사)

1999년 중앙대학교 컴퓨터공학과 (석사)

1999년~현재 중앙대학교 컴퓨터 공학과 박사과정

관심분야 : 광네트워크, 이동통신보안, 정보보호

여 상 수

e-mail : ssyeo@alg.cse.cau.ac.kr

1997년 중앙대학교 컴퓨터공학과 (학사)

1999년 중앙대학교 컴퓨터공학과 (석사)

2000년~현재 중앙대학교 컴퓨터 공학과 박사과정

관심분야 : 저작권 보호, 생물정보학

김 성 권

e-mail : skkim@cau.ac.kr

1981년 서울대학교 계산통계학과 졸업(학사)

1983년 한국과학기술원 전산학과 졸업(석사)

1983년~1985년 목포대학교 재직

1990년 University of Washington 전산학과(박사)

1991년~1996년 경성대학교 재직

1996년~현재 중앙대학교 컴퓨터공학과 재직중

관심분야 : 알고리즘, 컴퓨터이론, 정보보호