직병렬 유향 그래프의 삼차원 대칭성 탐지 알고리즘

홍 석희[○] Peter Eades
Dept. of Computer Science & Software Engineering, University of Newcastle
{shhong, eades}@cs.newcastle.edu.au

How to Find Three Dimensional Symmetry in Series Parallel Digraphs

Seok-Hee Hong[○] Peter Eades
Dept. of Computer Science & Software Engineering, University of Newcastle

요 약

대칭성(symmetry)은 그래프를 가시화하여 기하학적 표현을 구축하는 그래프 드로잉 분야에서 그래프의 구조와 특성을 명확하게 표현해주는 가장 중요한 평가 기준이다. 하지만 현재까지는 이차원 평면에서의 대칭성 문제에 대해서만 기존 연구가 이루어져왔을 뿐 해상도를 증가시키고 대칭성을 보다 풍부하게 표현할 수 있는 그래프의 삼차원 대칭 드로잉에 관한 연구는 아직 미약한 실정이다.

본 논문에서는 직병렬 유향 그래프에서의 삼차원 대칭성을 탐지하는 알고리즘을 제시하였다. 직병렬 유향 그래프는 소프트웨어 가시화나 네트워크 모델링 등에 자주 사용되는 유용한 그래프이다. 이 알고리즘은 직병렬 유향 그래프의 최대의 대칭성을 보여주는 삼차원 드로잉 알고리즘의 기반이 된다.

1. 서 론

대칭성은 그래프를 가시화하여 이차원 평면 또는 삼차원 공간상에 기하학적 표현을 구축하는 그래프 드로잉 분야에서 그래프의 구조와 특성을 시각적으로 표현하는 가장 중요한 평가 기준이다[1]. 또한 대칭 드로잉은 전체 그래프가 크기가 작은 동형의 부그래프들로부터 반복적으로 구성됨을 보여줌으로써 전체 그래프에 대한 이해를 명확하게 한다[6].

그래프의 대칭 드로잉에 관한 기존 연구는 이차원 평면상의 대칭성 문제에 대해서만 주로 연구되어 일반 그래프의 대칭성을 탐지하는 문제는 *NP-complete*로 증명되었으며 트리, 외부평면(outerplanar) 그래프, 임베딩된 평면 그래프의 이차원 대칭성을 탐지하는 알고리즘이 제시되었다[5,6,7]. 또한 직병렬 유향 그래프(series-parallel digraph)와 평면 그래프를 대상으로 이차원 대칭성을 탐지하는 알고리즘이 제안되었다[2,3].

그러나 최근 그래프의 삼차원 대칭성 모델이 제시되었으며 이를 기반으로 하여 트리의 최대의 삼차원 대칭성을 탐지하는 알고리즘이 제안되었다[9]. 본 논문에서는 삼차원 대칭 드로잉의 대상 그래프를 직병렬 유향 그래프로 확장하여 직병렬 유향 그래프에서 최대의 삼차원 대칭성을 탐지하는 알고리즘을 제안한다.

직병렬 유향 그래프는 데이터 플로우 다이아그램, 종속도 챠트(dependancy chart) 및 PERT 네트워크 등의 모델링에 자주 사용되는 유용한 그래프이다. 삼차원 대칭 드로잉은 해상도를 증가시키며 보다 풍부한 대칭을 표현할 수 있는 장점을 갖는다.

2. 직병렬 유향 그래프

직병렬 유향 그래프는 다음과 같이 정의된다[8].

[정의 1] 직병렬 유향 그래프

1. 두 개의 정점을 연결하는 하나의 유향 에지로 구성된 유향 그래프는 직병렬 유향 그래프이다.
2. $G_1, G_2, \ldots, G_k$가 직병렬 유향 그래프인 경우 다음의 연산에 의해 구성되는 그래프 G 역시 직병렬 유향 그래프이다.
   a. 직렬 복합(series composition) : G_i의 끝점(sink)과 G_{i+1}의 시작점(source)을 일치시킴으로써 G를 구성한다($1 \le i < k$).
   b. 병렬 복합(parallel composition) : G_i의 모든 시작점을 하나로 일치시키고 G_i의 각 끝점을 하나로 일치시킴으로써 G를 구성한다($1 \le i \le k$).

이 때 $G_1, G_2, \ldots, G_k$를 성분(component)이라고 한다.

직병렬 유향 그래프는 이진 분리 트리(binary decomposition tree)의 형태로 표현이 가능하다[8]. 그러나 이는 유일하지 않으므로 대칭성 탐지를 위해서는 이차원 대칭성 탐지시 정의되었던 정규 분리 트리(canonical decomposition tree) *CDT*를 사용한다[3]. 정규 분리 트리의 각 단말 노드는 직병렬 유향 그래프의 한 노드를 나타내고 비단말 노드들은 연산을 나타내기 위해 S 또는 P로 레이블링 되어 있다[3, 8].

3. 직병렬 유향 그래프의 삼차원 대칭성 탐지 알고리즘

삼차원 기하학적 물체의 대칭성의 종류는 rotation, reflection, inversion, rotatory inversion의 4가지로 분류가 가능하다[4]. 단 rotation은 축을 중심으로 한 회전이며 reflection은 면에 대한 reflection을 의미한다.

최근 그래프의 삼차원 대칭 드로잉을 구축하기 위한 삼차원 대칭성 모델이 제안되었다[9]. 그러나 직병렬 유향 그래프의 경우에는 보다 간단하게 삼차원 대칭성을 탐지할 수 있다. 이는 직병렬 유향 그래프에는 단 하나의 시작점과 단 하나의 끝점이 존재하므로 이들 두 정점은 서로 대응되거나 각각 자신으로 대응되어야 하기 때문이다. 따라서 직병렬 유향 그래프의 삼차원 대칭성의 종류는 다음과 같이 정의할 수 있다.

1. rotation(회전 대칭)
   1.1 z축을 중심으로 하는 k-fold 수직 대칭축
   1.2 xy 평면상에 존재하는 2-fold 수평 대칭축 k개
2. reflection(면 대칭)
   2.1 xy 평면상에 존재하는 수평 면대칭
   2.2 z축을 포함하는 k개의 수직 면대칭
3. inversion
   3.1 central inversion
   3.2 rotatory inversion

직병렬 유향 그래프의 이차원 대칭성에는 수직 대칭과 수평 대칭의 두 개의 축 대칭 및 180도 회전 대칭과 identity의 두 개의 회전 대칭, 단 4개의 대칭만이 존재했었다. 그러므로 삼차원 대칭성은 이차원 대칭성에 비해 그 수가 훨씬 증가한다.

직병렬 유향 그래프의 삼차원 대칭성을 탐지하기 전에 각 대칭성간의 관계를 살펴보기로 한다. 먼저 수직 회전 대칭과 수평 면 대칭이 존재하면 수평 회전 대칭이 존재한다. 또한 수직 회전 대칭과 inversion이 존재하면 rotatory inversion이 존재한다.

그러므로 수직 회전 대칭과 수평 면 대칭 그리고 inversion이 존재하는 지를 먼저 탐지하면 그 결과를 기반으로 나머지 대칭성들의 존재 여부 및 그 수를 계산할 수 있다.

따라서 본 논문에서는 수직 회전 대칭과 수평 면 대칭 그리고 inversion을 탐지하는 알고리즘을 제안한다. 직병렬 유향 그래프의 삼차원 대칭성 탐지 알고리즘은 다음과 같이 기술된다.

알고리즘 3DSymmetry_SPDigraph

1. 주어진 직병렬 유향 그래프의 정규 분리 트리 *CDT*를 구성한다.
2. 동형성 집합 트리 *ICT*를 구성하고 각 노드 v마다 n_v, L_v 값을 계산한다.
3. 기본이 되는 세 가지 대칭성을 각각 탐지한다.
   3.1 **알고리즘 Vertical_Labelling**
   알고리즘 Vertical_Rotation
   3.2 **알고리즘 Horizontal_Labelling**
   알고리즘 Horizontal_Reflection
   3.3 **알고리즘 Inversion_Labelling**
   알고리즘 Inversion
4. 단계 3의 결과를 기반으로 나머지 대칭성을 분석한다.

동형성 집합 트리(Isomorphism Class Tree) *ICT*는 트리의 삼차원 대칭성 탐지 알고리즘에서 제안된 트리로 각 노드 v는 동형성 집합을 의미한다[9]. 또한 각 노드마다 그 동형성 집합의 크기 n_v와 그에 속한 부트리의 가능한 대칭 드로잉의 대칭의 개수를 리스트로 구성한 L_v를 유지하고 있다.

*ICT*와 *CDT* 두 트리를 기반으로 하여 직병렬 유향 그래프의 삼차원 대칭성을 탐지할 수 있다. 이는 다음 각 절에서 설명한다.

3.1 알고리즘 Vertical_Rotation

직병렬 유향 그래프의 경우에는 각 성분마다 최대 하나의 부트리를 수직 대칭축 상에 배치 가능하므로 이를 계산하기 위해서는 *ICT*의 각 노드에 저장된 정보를 사용한다.

직병렬 유향 그래프가 하나의 성분으로만 구성된 경우에는 트리의 삼차원 대칭성 탐지 알고리즘 중에서 하나의 부트리가 축 상에 위치하는 경우의 알고리즘을 적용하여 최대의 k-fold 수직 대칭을 계산할 수 있다. 두 개 이상의 성분으로 구성된 경우에는 각 성분의 대칭성으로 리스트를 구성하여 그 리스트의 원소들로 구성되는 최대의 *gcd*를 계산한다.

대칭성을 탐지하기 전에 먼저 *CDT*를 레이블링하는 작업이 필요하다. 이는 *CDT*의 각 노드 u마다 u의 동형성 코드를 나타내는 정수 $code(u)$와 u의 자식 노드들의 동형성 코드들로 구성된 리스트 $tuple(u)$를 계산하는 작업이다. **알고리즘 Vertical_Labelling**은 다음과 같다.

알고리즘 Vertical_Labelling

1. *CDT*의 각 노드에 레벨을 할당한다.
2. *CDT*의 각 단말 노드 u의 $tuple(u)$를 0으로 초기화한다.
3. *CDT*의 최하위 레벨부터 루트 레벨까지 각 레벨 i에 대하여 다음을 반복한다.
   3.1 레벨 i의 비단말 노드 u에 대해 u의 자식 노드들을 순서대로 $v_1, v_2, \ldots, v_k$ 라 하면,
   $tuple(u) = (code(v_1), code(v_2), \ldots, code(v_k))$
   3.2 u가 P 노드인 경우 $tuple(u)$를 정렬한다.
   3.3 레벨 i에 위치하는 노드들의 *tuple*들로 구성된 리스트 L을 구성하고 이를 사전순서로 정렬한다.
   3.4 레벨 i의 각 노드 u 마다, 처음 나타나는 *tuple*에 0을 할당하고 그 다음에 나타나는 상이한 *tuple*에 1씩 증가한 값을 할당하여 $code(u)$를 계산한다.

*CDT*의 루트 노드 r과 그에 연결된 자식 노드들 $v_1, v_2, \ldots, v_l$이라 하고 *ICT*의 루트 노드에 연결된 자식 노드들을 $s_1, s_2, \ldots, s_k$라 하자. **알고리즘 Vertical_Rotation**은 다음과 같다.

알고리즘 Vertical_Rotation(r)

1. if (r이 P 노드인 경우) then
   /* 직병렬 유향 그래프가 하나의 성분으로 구성된 경우 */
   1.1 $g = gcd(s_1, s_2, \ldots, s_k)$
   $L_r \leftarrow g$
   1.2 for i = 1 to k

$g_i = gcd(s_1, s_2, ..., s_i - 1, ..., s_k)$
for i = 1 to k
for each e_j in Ls_i
$g_i' = gcd(g_i, e_j)$
if ($g_i' \neq 1$) and ($g_i' \notin Lr$)
then $Lr \leftarrow g_i'$
1.3 $g = max(Lr)$
1.4 return(Lr, g)
2. if (r이 S 노드인 경우) then
/* 두 개 이상의 성분으로 구성된 경우 */
2.1 r의 자식 노드 $v_1, v_2, ..., v_l$ 각각에 대하여 **Vertical_Rotation**(v_i)를 적용한다.
2.2 for i = 1 to l do
for j = $max(n_i)$ downto 1
if (j가 모든 L_i에 대해 원소 e_i를 나누는 경우)
then exit
2.3 return(j)

3.2 알고리즘 Horizontal_reflection

알고리즘 Horizontal_Labelling은 *CDT*의 절반에 대하여 *S* 노드의 자식 노드의 순서를 역순으로 바꾸어 **알고리즘 Vertical_Labelling**과 동일한 방법으로 레이블링하므로 자세한 알고리즘의 기술은 생략한다. 삼차원 수평 면 대칭을 탐지하는 알고리즘은 다음과 같다.

알고리즘 Horizontal_Reflection(r)
1. if (r이 P 노드인 경우) then
if (r의 자식 노드 $v_1, v_2, ..., v_l$ 각각에 대하여 **Horizontal_Reflection**(v_i)가 *true*)
then return(*true*)
else return(*false*)
2. if (r이 S 노드인 경우) then
2.1 if (*tuple(r)*이 좌우대칭(palindrome)이 아닌 경우)
then return(*false*)
2.2 if (*tuple(r)*이 좌우대칭이고 길이가 짝수인 경우)
then return(*true*)
2.3 if (*tuple(r)*이 좌우대칭이고 길이가 홀수인 경우)
then return(**Horizontal_Reflection**(v_i))
단, v_i는 좌우대칭의 가운데 성분이다.

3.3 알고리즘 Inversion

알고리즘 Inversion_Labelling은 **Horizontal_Labelling**과 유사하므로 자세한 알고리즘의 기술은 생략한다. inversion을 탐지하는 알고리즘은 다음과 같다.

알고리즘 Inversion(r)
1. if (r이 P 노드인 경우) then
1.1 r의 각 자식 노드 $v_1, v_2, ..., v_l$을 *tuple*(v_i)의 역순으로 짝지운다.
1.2 if (두 개 이상의 성분이 짝을 이루지 못하는 경우)
then return(*false*)
1.3 if (모든 v_i가 짝지워지는 경우)
then return(*true*)
1.4 if (단 하나의 v_i가 짝을 이루지 못하는 경우)
then return(**Inversion**(v_i))
2. if (r이 S 노드인 경우) then
2.1 if (*tuple(r)*이 좌우대칭이 아닌 경우)
then return(*false*)
2.2 if (*tuple(r)*이 좌우대칭이고 길이가 짝수인 경우)
then return(*true*)
2.3 if (*tuple(r)*이 좌우대칭이고 길이가 홀수인 경우)
then return(**Inversion**(v))
단, v는 좌우대칭의 가운데 성분이다.

[정리 1] 직병렬 유향 그래프의 삼차원 대칭성을 탐지하는 알고리즘은 $O(n^2)$ 시간이 소요된다(단 n은 정점의 개수이다).

4. 결 론

대칭성은 그래프 드로잉시 그래프의 구조와 특성을 명확하게 보여주는 가장 중요한 미적 기준이다. 특히 삼차원 대칭 드로잉은 이차원에 비해 대칭의 수가 크게 증가하므로 풍부한 대칭성의 표현을 가능하게 해준다.

본 논문에서는 직병렬 유향 그래프에서 삼차원 대칭성을 탐지하는 알고리즘을 제안하였다. 이 알고리즘은 직병렬 유향 그래프의 최대의 대칭성을 보여주는 삼차원 드로잉 알고리즘 개발의 기반이 된다.

향후 연구과제로는 보다 다양한 그래프를 대상으로 그의 삼차원 대칭 드로잉을 구축하는 알고리즘을 개발하는 것이다.

참고문헌

[1] G. Di Battista, P. Eades, R. Tamassia and I. G. Tollis, Graph Drawing : Algorithms for the Visualization of Graphs, Prentice-Hall, 1998.
[2] S-H. Hong, P. Eades and S-H. Lee, "Finding Planar Geometric Automorphisms in Planar Graphs," Algorithms and Computation(Proc. ISAAC'98), LNCS vol. 1533, pp 277-286, Springer Verlag, 1998.
[3] S-H. Hong, P. Eades, A. Quigley and S-H. Lee, "Drawing Algorithms for Series-Parallel Digraphs in Two and Three Dimensions," Graph Drawing (Proc. GD'98), LNCS vol. 1547, pp. 198-209, Springer Verlag, 1998.
[4] E. H. Lockwood and R. H. Macmillan, Geometric Symmetry, Cambridge University Press, Cambridge, 1978.
[5] J. Manning and M. J. Atallah, "Fast Detection and Display of Symmetry in Trees," Congressus Numerantium, 64, pp. 159-169, 1988.
[6] J. Manning, "Geometric Symmetry in Graphs," Ph.D. Thesis, Purdue Univ., 1990.
[7] J. Manning and M. J. Atallah, "Fast Detection and Display of Symmetry in Outerplanar Graphs," Discrete Applied Mathematics, 39, pp. 13-35, 1992.
[8] J. Valdes, R. Tarjan and E. Lawler, "The Recognition of Seriea-Parallel Digraphs," SIAM Journal on Computing 11(2), pp. 298-313, 1982.
[9] 홍 석희, Peter Eades "트리의 삼차원 대칭성 탐지 알고리즘," 2000 한국 정보과학회 춘계학술발표, *submitted*