내포 병렬성을 가진 공유메모리 프로그램의 3차원 시각화

박명철* · 허화라* · 하석운**

The 3-Dimensional Visualization in Shared-Memory Programs with Nested Parallelism

Myeong-chul Park* · Hwa-ra Hur* · Seok-wun Ha**

요 약

내포 병렬성을 가지는 병렬 프로그램은 동기화 없이 병행적으로 수행되는 양상으로 인하여 비결정적인 결과를 초래하는 경향이 있다. 이러한 오류를 탐지하기 위하여 다양한 시각화 기법이 이용되고 있지만, 공간의 제한성과 과다한 추상화로 인하여 직관성이 매우 저하되는 실정이다. 본 논문에서는 내포 병렬성을 가지는 복잡한 병렬 프로그램의 전역적 구조를 사용자에게 제공하는 3차원 시각화 엔진을 제안한다. 제안된 시각화 엔진은 전역적 구조를 사용자에게 제공함으로서 프로그램의 이해를 용이하게 하고 효과적인 디버깅 환경을 제공한다.

ABSTRACT

A parallel program including a nested parallelism has a result of non-deterministic because of executed concurrently without synchronization. In order to detect like this error the visualization technique which is various is used. But the intuition characteristic is decreased because of limits of space and excessive abstraction. In this paper, proposes 3-D visualization engines which provide global structure of the arranging in a parallel program with nested parallelism which is complicated to the user. The visualization engine which is proposed provides global structure to the user as program easily to understand, it provides an effective debugging environment.

키워드

Visualization, Parallel Program, OpenMP, Labeling

Ⅰ. 서 론

OpenMP와 같은 공유메모리를 사용하는 병렬 프로그램은 복잡한 수행양상과 스레드간의 비동기적인 행위로 인하여 비결정적인 결과를 초래한다.[1] 이는 내포 깊이가 큰 프로그램에서는 그 심각성이 한층 더 높다.[2,3] 이러한 문제를 해결하기 위한 방법으로 사용자에게 복잡한 수행양상을 시각정보를 이용하여 디버깅 환경을 제공하는 다양한 시각화 엔진이 소개되고 있으나, 시각화 공간의 제한성과 과다한 스레드 추상화로 인하여 효과적인 시각화를 제공하지 못하고 있다.[4]-[6] 본 논문에서는 병렬 프로그램의 시각화를 위하여 수행양상의 전역적인 구조와 직관력 높은 3차원 시각화 도구를 제안한다. 3차원 시각화는 디스크 구조를 이용하여 내포성을

* 송호대학 컴퓨터정보과
** 국립 경상대학교 컴퓨터과학과(교신저자)

접수일자 : 2007. 8. 23

쉽게 식별할 수 있고, 디스크의 확장과 은닉으로 쉽게 추상화 기능을 통한 공간 제약성을 해결한다. 시각화를 위한 스레드 정보와 사건 접근정보는 기존 연구에서 소개한 eNR 레이블링 방법을 이용하였다.[7]

본 논문에서는 3차원 시각화에 적합한 변형된 레이블링 방법을 소개하고, 내포 병렬성을 식별할 수 있는 디스크 뷰 구조를 보인다. 스레드의 클러스터링을 위한 백본 디스크와 시각화 도구와 사용자 간의 상호작용성을 높이기 위하여 다양한 인터페이스 환경을 제공한다. 제안하는 도구는 플랫폼에 제약이 없고 공개되어 있는 OpenGL을 사용하여 구현되었다. OpenGL은 그래픽 라이브러리의 산업화 표준으로서 이식이 쉽고 속도가 빠른 장점을 가진다. 본 논문의 2장에서는 공유 메모리 기반의 병렬프로그램 API인 OpenMP 프로그램과 기존의 시각화 방법에 대해 살펴보고, 3장에서는 본 논문에서 제한하는 3차원 디스크 뷰의 구조와 실제 도구의 구현을 보인다. 그리고 4장에서는 결론과 향후 연구를 밝힌다.

II. 연구배경

2.1 OpenMP 프로그램과 레이블링 방법

OpenMP는 공유메모리를 사용하는 병렬프로그램의 위한 API로써 다양한 디렉티브와 라이브러리를 제공하고 있다. 일반적으로 스레드의 생성(*Fork*)과 합류(*Join*)을 통하여 프로그램이 구성된다. 이는 그림1과 같이 가장 일반화된 2차원 시각화 방법인 방향성이 있는 비순환 그래프인 POEG(Partial Order Execution Graph)으로 설명할 수 있다.[8] 그림 1에서 *Ta* 스레드의 *Fork*연산에 의해 생성된 스레드가 *Tb*와 *Tc*가 된다. 이때 *Tb*, *Tc*는 *Ta*에 비해 내포 깊이가 1증가하게 된다. 두 스레드는 *Tg* 시점에서 *Join* 연산에 의해 병렬 수행을 마치게 된다. 수행상의 스레드을 식별하기 위해서는 각 스레드의 고유 정보인 레이블 정보가 필요하다. 기존 연구에서 제안한 eNR 기법은 2차원 시각화를 위하여 제안된 방법이므로 3차원 시각화를 위해서 수정이 요구된다. POEG 형태의 표현 방법들은 수행의 부분적인 순서화를 표현하는 것으로는 적합하지만 전역적인 수행 양상을 보이지는 못하고 있다. 또한 복잡도가 높고 방대한 자료를 2차원에 표현하기에는 공간적인 부족함이 유발된다.[9] 그림 1의 OpenMP 프로그램에 대한 POEG을 그림 2에서 보이고 있다. 이렇게 POEG은 병렬 프로그램의 부분적인 실행순서를 나타내며 각 스레드간의 발생후 관계를 표현한다.

```
#pragma omp parallel for shared(S)
  for(i=0;i<2;i++) {
    if(i==0) S++;
    else {
     #pragma omp parallel for shared(S)
      for(j=0;j<3;j++) {
       y++;
       S = S - y;
      }
    }
  ...
```

그림 1. OpenMP프로그램의 예
Fig. 1 Example of OpenMP Program

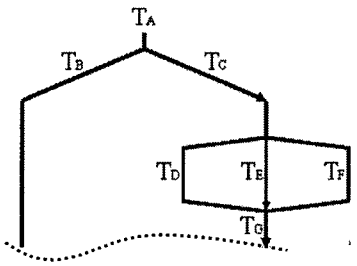

그림 2. POEG 구조상의 표현
Fig. 2 POEG(Partial Order Execution Graph)

POEG의 각 정점은 스레드의 생성과 합류를 의미하고 정점간의 간선은 순서적인 흐름을 의미한다. *Tb*와 *Tc*는 병행성 관계임을 알 수 있고, *Tc*와 *Tg*는 순서화 관계임을 알 수 있다. 병행적으로 수행되는 스레드의 양상을 시각적으로 표현하기 위해서는 수행되는 각 스레드의 식별정보가 필요한데, 이를 위해 사용되는 일련의 요소들을 레이블링이라 한다.[3,7] 본 논문에서는 eNR 레이블링 기법을 변형하여 3차원 시각화에 적합한 새로운 레이블링 기법을 사용한다. 생성되는 레이블 정보의 인자는 표 1과 같다.

표 1. 레이블 정보의 인자
Table 1. Parameters of Label Information

인 자	의 미
J_t	현재 Task 까지의 조인횟수
N_{dep}	현재 Task의 내포깊이
(Xs,Xe)	현재 Task의 x축 시작, 끝좌표
(Ys,Ye)	현재 Task의 y축 시작, 끝좌표
B_{tot}	Fork된 전체 Task의 수
B_{no}	B_{tot}에 따른 순서 번호

표 1의 표현 인자에 의해 형성되는 레이블 표현은 아래와 같다.

$$[\,J_t, N_{dep}, (Xs, Xe), (Ys, Ye), B_{tot}, B_{no}\,]$$

기존의 eNR 레이블링 기법을 기준으로 하여 3차원 디스크 뷰을 생성하기 위한 정보로 B_{tot}와 B_{no}을 레이블링 정보에 추가하였다. B_{tot}는 *Fork* 연산 시에 생성되는 하위 스레드의 개수를 의미하고, B_{no}는 생성되는 각 스레드의 순서 번호를 의미한다. 이렇게 생성된 Ta의 레이블링 정보는 다음과 같다.

$$Ta \text{ Label}=[0,0,(1,MaxInt),(1,MaxInt),0,0]$$

여기서, *MaxInt*는 시스템에서 표현할 수 있는 최대 정수값이다. 아래는 레이블 정보를 생성하기 위한 알고리즘을 보이고 있다.

Algorithm[Init]:

$Jt := 0$
$Ndep := 0$
$Xs := 1$
$Xe := MaxInt$
$Ys := 1$
$Ye := MaxInt$
$Btot := 0$
$Bno := 0$

Algorithm[Fork]:

if $Ta(Ndep)$ is odd

$T\beta(Xs) = Ta(Xs)+(N-I)\times Vx(Ta)/N$

$$T\beta(Xe) = \begin{cases} T\beta(Xs) + Vx(Ta)/N-1 & \text{if } I<N \\ Ta(Xe) & \text{otherwise} \end{cases}$$

$T\beta(Ys) = Ta(Ys)$
$T\beta(Ye) = Ta(Ye)$

else

$T\beta(Xs) = Ta(Xs)$
$T\beta(Xe) = Ta(Xe)$
$T\beta(Ys) = Ta(Ys)+(N-I)\times Vy(Ta)/N$

$$T\beta(Ye) = \begin{cases} T\beta(Ys) + Vy(Ta)/N-1 & \text{if } I<N \\ Ta(Ye) & \text{otherwise} \end{cases}$$

end if
$T\beta(Btot) = N$
$T\beta(Bno) = I$
$T\beta(Jt) = Ta(Jt)$
$T\beta(Ndep) = Ta(Ndep) + 1$
where $I=N, N-1, N-2,\ldots,1$ (N is number of Fork)

Algorithm[Join]:

$T\gamma(Xs) = \min\{Ta(Xs), T\gamma(Xs)\}$
$T\gamma(Xe) = \min\{Ta(Xe), T\gamma(Xe)\}$
$T\gamma(Ys) = \min\{Ta(Ys), T\gamma(Ys)\}$
$T\gamma(Ye) = \min\{Ta(Ye), T\gamma(Ye)\}$
$T\gamma(Jt) = \min\{Ta(Jt), T\gamma(Jt)\}$
$T\gamma(Ndep) = Ta(Ndep) - 1$
$T\gamma(Btot) = 0$
$T\gamma(Bno) = 0$

기존의 eNR 레이블링 알고리즘과 추가된 정보를 이용하여 그림1의 프로그램에 대한 레이블링 정보는 표2와 같다. (단, MaxInt = 50 가정)

표 2. 레이블이 붙여진 병렬 스레드
Table 2. Labeled Parallel Threads

Tasks	Label Information
T_A	[0,0,(1,50),(1,50),-,-]
T_B	[0,1,(1,25),(1,50),2,1]
T_C	[0,1,(26,50),(1,50),2,1]
T_D	[0,2,(26,50),(1,16),3,1]
T_E	[0,2,(26,50),(17,33),3,2]
T_F	[0,2,(26,50),(34,50),3,3]
T_G	[1,1,(26,50),(1,50),-,-]

2.2 기존의 시각화 도구의 문제점

병렬 프로그램의 이해와 디버깅을 위한 시각화방법은 다양한 형태로 제안되어 왔다. 그러나 대부분의 기법들이 2차원 공간 내에서 부분적인 스레드의 수행양상만을 제공하기 때문에 공간 제약성과 전역적인 수행양상을 보이는 데는 한계가 있다.[10] 3차원 공간상에 표현하는 기법도 소개되었지만, 과다한 추상화로 인하여 프로그램 이해의 직관력을 저해하는 경향을 보여 왔다.[11]

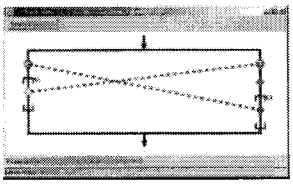

그림 3. 확장적 시각화[10]
Fig. 3 Scalable Graph[10]

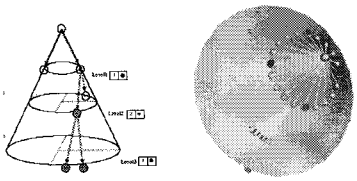

그림 4. 콘트리 시각화[11]
Fig. 4 Cone Visualization[11]

Ⅲ. 3차원 시각화 도구의 구현

3.1 3차원 디스크 뷰의 구조

기본 연구에서 제안한 2차원 평면상의 영상화는 영상이 가지는 색상정보를 이용하여 병행성을 판별하기 위함이 중요한 목적이였다.[7] 그러나, 이는 프로그램의 수행양상을 사용자에게 보이기에는 직관력을 저해하는 제한점을 가지고 있다. 그래서 사용자에게 병렬 프로그램의 수행양상을 전역적이면서 직관력 높게 보여주기 위한 방법으로 본 연구에서는 3차원 디스크 뷰를 제공한다. 그림 5에 보는 것과 같이 3차원 뷰는 몇가지 특징적인 객체를 가진다.

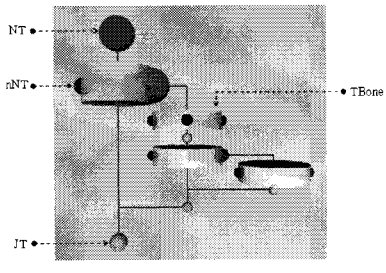

그림 5. 3차원 시각화 뷰
Fig. 5. 3D Visualization View

여기서, *NT*라 명명되는 객체는 내포스레드를 가지는 스레드를 표시하는 것이고, *nNT*라 명명된 객체는 내포스레드가 없는 독립된 스레드를 표시한 것이다. 그리고 *JT*는 *Join*된 이후의 스레드를 표시하며 *TBone*은 병행성을 가지고 있는 형제 스레드를 표시하는 백본 디스크이다. 백본 디스크상의 스레드의 배치는 레이블링 정보의 B_{tot} 정보를 이용하여 균등분할하여 배치된다. 배치된 스레드중에서 *NT*스레드가 사용자에 의해 활성화가 되면, 디스크의 외곽방향으로 회전하여 새로운 백본 디스크를 생성하게 된다. 동시의 동일 디스크의 *NT* 스레드가 활성화가 되면 상위 디스크의 반경이 늘어나면서 하위 디스크를 보이게 된다. 일반적으로 사용자는 두 사건간의 병행성관계나 순서화 관계를 식별하기 위한 목적으로 스레드를 활성화 시키므로 디스크 반경 상의 제한은 큰 문제가 되지 않을 것이다. 또한 스레드 추상화를 이용하여 사용자의 관심 스레드 외에는 백본 디스크를 디스플레이 하지 않고 *NT*스레드 상태로 표현되기 때문에 기존의 복잡한 스레드 표현은 줄어들 것이다. 만약, 사용자가 전체 스레드의 확장된 모습을 보고자 할 때는 축소, 확대, 회전, 백본 은닉등의 다양한 사용자 인터페이스을 이용하여 효과적으로 표현하게 된다. 본 논문에서 제안한 기법의 성능을 검증하고 그 결과를 보이기 위해 Windows 환경에서 C++/OpenGL을 사용하여 도구를 구현하였다.

3.2 스레드의 추상화

스레드 추상화는 동일한 내포 수준의 스레드는 동일 디스크상에 시각화하는 것을 원칙으로 한다. NT객체는 내포 스레드를 가지는 스레드 객체이다. 스레드의 생성이 많을 경우 상당히 복잡한 시각화 양상을 보이므로 이때는 사용자로 하여금 추상화 수준을 높여 전역적이 구조만을 보일 수 있는 방법을 제공한다. 붉은 색의 *NT*객체가 정적으로 움직이지 않으며(애니메이션에 의한 정지상태), 이미 내포 디스크를 보인 상태이고 동적으로 움직이면(애니메이션에 의한 모션 상태) 내부에 추상화된 은닉 디스크가 존재함을 표시한다. 이러한 추상화는 보다 확장적인(Scalable) 시각화를 도모할 수 있다. 이렇게 함축된 스레드의 예를 그림 6에서 보이고 있다. 그림 6의 (a)에서 오른쪽 상단의 붉은 *NT*객체를 클릭하면, (b)와 같이 함축된 스레드가 표시되게 된다.

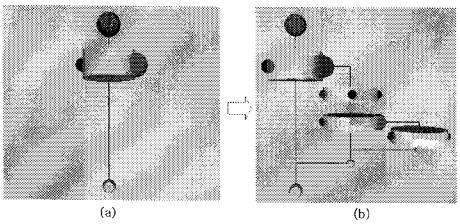

그림 6. 스레드의 추상화
Fig. 6. Abstraction of Thread

3.3 시각화의 상호작용성

기존 시각화 도구의 대부분은 사용자의 입력을 실시간으로 시각화에 적용하지는 못하고 있다. 특히, 3차원 시각화의 경우 공간좌표를 사용하기 때문에 사용자에게 보여지는 정면 뷰 외에도 감춰진 객체가 백그라운드 영역에 존재할 수 있다. 하지만, 기존의 정적인 특성을 가진 시각화 도구는 이러한 백그라운드 영역에 존재하는 객체를 보이기에는 제한적이다. 또한, 시각적 효과를 극대화시키기 위한 다양한 사용자 인터페이스를 제공하지 못하고 있다. 본 연구에서 제안 하는 시각화 엔진은 그림 7과 같이 사용자의 다양한 요구사항을 상호작용적으로 표현할 수 있는 인터페이스를 제공한다. 이는 사용자로 하여금 보다 직관적으로 프로그램 수행을 판단할 수 있게 도와준다. 그림 7의 (a)는 표준적인 뷰를 보이고 있고, (b)와 (c)는 사용자에 의해 확대, 축소된 뷰를 보이고 있다. 그리고 (d)는 복잡한 스레드의 3차원 구조를 사용자가 자유롭게 회전시키면서 표현되는 예를 보이고 있다.

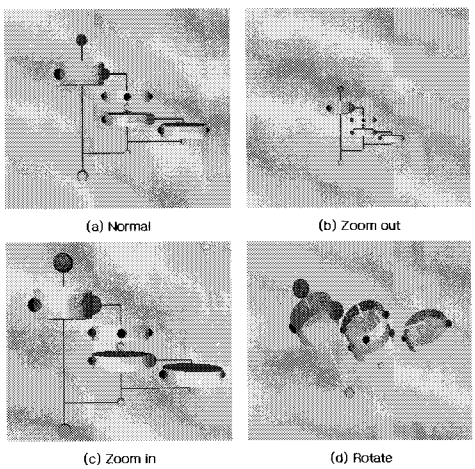

그림 7. 3차원 뷰의 상호작용
Fig. 7. Interactions of 3D View

그림 8은 기존의 연구에서 제안한 2차원 영상 시각화 엔진과 본 연구에서 제안하는 도구를 결합시킨 도구 화면이다. 2차원 영상 정보에서 해당 스레드를 클릭하게 되면, 3차원 디스크 뷰상에 해당 스레드가 표시되고 하단에 레이블링 정보와 프로그램의 소스 라인이 표시되게 된다.

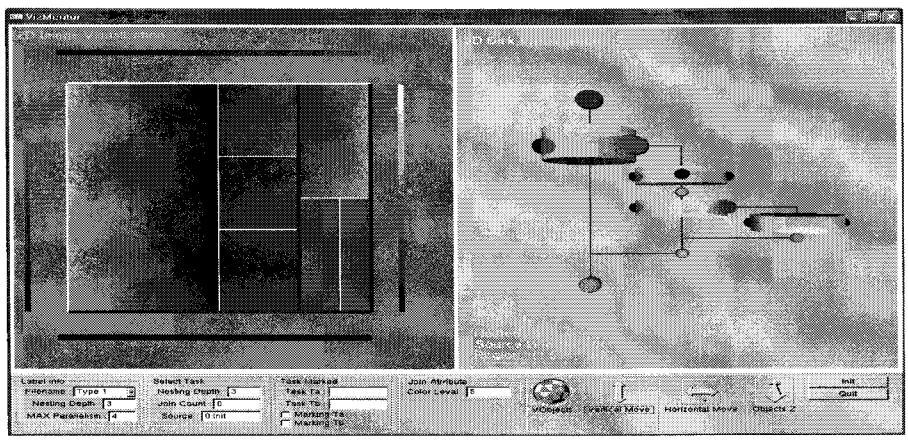

그림 8. 통합 시각화 뷰
Fig. 8. Visualization View

Ⅳ. 결론 및 향후연구

본 논문은 내포 병렬성을 가지는 공유메모리 프로그램의 3차원 시각화 방법을 제안하였다. 병렬 프로그램의 복잡한 수행양상의 전역적인 수행구조를 보이는 동시에 사용자로 하여금 직관적으로 스레드들 간의 관계를 인지할 수 있는 디스크 뷰를 제공하였다.제공된 도구는 스레드 정보에 해당하는 소스 코드를 보임으로서 프로그램을 이해하는데 도움이 되며, 디버깅 환경에 활용될 수 있을 것이다. 향후에는 시각화 엔진과 함께 병렬 프로그램의 정적 분석도구와 동적 분석도구를 개발하여 연계시키는 연구가 이루어 질 것이다.

참고문헌

[1] Audenaert, K., "Maintaining Concurrency Information for On-the-fly Data Race Detection," *Int'l Conf. on Parallel Computing : Fundamentals, Applications and New Directions*, Bonn, Germany, pp.319-326, Sept., 1997, Advances in Parallel Computing, Elsevier, North-Holland, Amsterdam, Vol.12, 1998.

[2] Kuhn, B., P. Petersen, and E. O'Toole, "OpenMP versus Threading in C/C++," EWOMP '99, Lund, Sweden, Sept. 1999.

[3] Netzer R. H. B., and B. P. Miller, "What are Race Condition? Some Issues and Formalization," *Letters on Programming Lang. and System*, 1(1):74-88, ACM, 1992.

[4] Dagum, L. and R. Menon, "OpenMP : an Industry-Standard API for Shared-Memory Programming," *Computational Scince and Engineering*, 5(1), IEEE, 46-55, Jan.-March, 1998.

[5] G. Kim, Y. Kim and Y. Jun, "Effective Race Visualization for Debugging OpenMP Programs," *Proc. 31nd KISS Conference*, Vol. 31 No. 02 pp. 13-15 Oct. 2004.

[6] Helmbold, D. P., C. E. McDowell, and J. Wang, "TraceViewer: A Graphical Browser for Trace Analysis," TR-90-59, UCSC. 1990

[7] 박명철, 하석운 "영상정보를 이용한 병렬 프로그램 내의 병행성 판별", *한국해양정보통신학회논문지*, 10권 12호 pp.2132-2139, 2006

[8] Citron D., D. G. Feitelson, and I. Exman, "Parallel Activity Roadmaps," *Int'l Conf. on Parallel Computing(ParCo '93)*, Parallel Computing: Trends and Applications pp. 593-596, Elsevier Science, 1994.

[9] Zernik, D., M. Snir, and D. Malki, "Using Visualization Tools to Understand Concurrency," *Software*, 9(3): 87-92, IEEE, May 1992.

[10] Kim, J., D. Kim, and Y. Jun, "Scalable Visualization for Debugging Races in OpenMP Programs," *Proc. of the 3rd Int'l Conf. on Communications in Computing (CIC)*, pp.259-265, Las Vegas, Nevada, June 2002.

[11] 임재선 "*OpenMP Program의 경합 탐지를 위한 3차원 원추상의 확장적 사건 시각화*", 경상대학교 석사학위논문, 2006

저자소개

박 명 철(Myeong-Chul Park)

2007년 경상대학교 컴퓨터과학과 (공학박사)
2007년 송호대학 컴퓨터정보과 전임강사

※관심분야 : 시각화, 임베디드소프트웨어, 병렬프로그래밍

허 화 라(Hwa-ra Hur)

2001년 부산대학교 전자공학과 (공학박사)
2000년 송호대학 컴퓨터정보과 부교수

※관심분야 : Time-Delay, 모델예측제어, 원격제어로봇

하 석 운(Seok-Wun Ha)

1995년 부산대학교 전자공학과 (공학박사)
1993년 경상대학교 컴퓨터과학과 교수

※관심분야 : 신경회로망, 컴퓨터비젼, 임베디드 시스템