제18회 한국정보처리학회 추계학술발표대회 논문집 제9권 제2호 (2002.11)

그리드 환경에서의 Computational Steering System에 관한 연구

구기범*, 박형우*, 이상산*
*한국과학기술정보연구원 슈퍼컴퓨팅센터
e-mail: {voxel,hwpark,sslee}@hpcnet.ne.kr

Study on Construction of Computational Steering Systems for Grid Environment

Gee-Bum Koo*, Hyoungwoo Park*, Sangsan Lee*
*KISTI Supercomputing Center

요 약

Computational steering system은 계산 및 응용 과학자들이 컴퓨터를 이용해서 보다 효과적이고 효율적인 방법으로 시뮬레이션을 진행하고 제어하기 위해서 제안되었다. 하지만 시간이 흐를수록 시뮬레이션의 규모가 커지면서 단일 컴퓨터 시스템으로 시뮬레이션을 수행하는 것이 어렵게 되었다. 이 문제를 해결하기 위해서 최근 새로운 형태의 슈퍼컴퓨팅 환경으로 주목받기 시작한 그리드와 computational steering system을 연계하는 방법에 대한 연구가 진행되고 있다. 본 논문에서는 computational steering system의 개념과 함께 그리드에서 운용 가능한 대표적인 computational steering system을 소개하고 KISTI 슈퍼컴퓨팅센터에서 구축하고 있는 국가 그리드 기반인 N* Grid를 위한 computational steering system의 구현 방향을 제시하고자 한다.

1. 서론

최근 고성능 컴퓨팅 분야에서는 그리드(Grid)가 새로운 형태의 컴퓨팅 환경으로 주목받기 시작했으며, 그에 대한 연구가 활발하게 진행되고 있다. 그리드란, 이기종의 컴퓨팅 자원과 대용량 저장장치, 그리고 다양한 과학 실험 장비를 하나의 네트워크로 연결하여 구성한 통합환경을 말한다.[1]

Computational steering system은 응용과학자들이 컴퓨터를 이용한 시뮬레이션을 보다 효율적이고 효과적인 방법으로 진행하고 제어할 수 있도록 지원하기 위해서 제안되었다.[2] 그러나 시간이 흐르면서 시뮬레이션의 규모가 단일 컴퓨터 시스템으로는 소화해낼 수 없을 정도로 커지게 되었다. 이를 위해서 그리드와 computational steering system을 접목하는 것이 새로운 해결책으로 제시되고 있다.

본 논문은 다음과 같이 구성되어 있다. 2장에서는 계산 그리드를 중심으로 그리드의 개념 및 특징을 소개한다. 3장에서는 computational steering system을 정의하고 그것이 갖는 특징에 대해 살펴볼 예정이다. 4장은 기존의 computational steering system들의 특징을 분석하고, 마지막 5장에서는 현재 KISTI 슈퍼컴퓨팅센터에서 계획중인 computational steering system의 구현 방향에 대해서 간략하게 소개하겠다.

2. 그리드의 개념

그리드는 사용 목적에 따라서 그림 1과 같이 나누어질 수 있다.[1]

계산 그리드(computational grid)는 많은 수의 컴퓨터를 네트워크로 연결해서 일반적인 컴퓨터가 갖는 계산 능력을 능가하는 고성능 컴퓨팅 자원을 제공하기 위해 구성된다. 계산 그리드는 다시 HPC (High Performance Computing) 그리드와 HTC (High Throughput Computing) 그리드로 나누어서

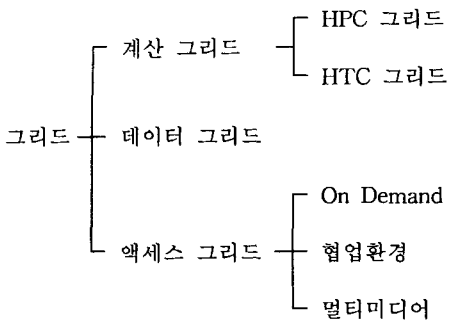

그림 1 그리드의 종류

생각할 수 있는데, HPC 그리드는 한 순간에 최대한 많은 컴퓨팅 자원을 동원해서 작업의 전체 수행시간을 줄이는 데에 초점을 맞추며, HTC 그리드는 주어진 시간 내에 가능한 많은 수의 작업을 처리하는 데에 중점을 둔다.

데이터 그리드는 지역적으로 분산된 방대한 자료의 통합 관리를 지원하기 위한 그리드이다. 이것은 고성능 컴퓨터보다는 대용량 저장장치를 연계하는 것에 초점을 맞추고 있으며 대량의 데이터를 다룰 필요가 있는 분야에 주로 사용된다. 과거에는 계산 그리드와 데이터 그리드를 따로 구분하는 경향이 있었으나 시간이 흐를수록 그 둘 사이의 경계선이 점점 모호해지는 추세이다.

액세스 그리드는 지역적으로 떨어져 있는 사람들이 가상공간에서의 상호작용(interaction)을 가능하도록 해주기 위한 그리드이다.

3. Computational Steering System의 소개

이제 응용 및 순수과학 분야 연구자들이 컴퓨터를 이용해서 대규모 시뮬레이션이나 과학 실험을 수행하는 것은 그리 드물지 않은 일이 되었다. 컴퓨터를 이용한 시뮬레이션을 단계별로 살펴보면 대략 다음과 같은 작업들로 구성되어 있음을 알 수 있다.

- 실험에 필요한 데이터와 파라메터 입력 및 수정
- 시뮬레이션 프로그램의 실행
- 결과 가시화

기존의 컴퓨터를 이용한 시뮬레이션은 위의 과정을 순서대로 반복하는 식으로 진행되었다. 즉, 사용자는 인터페이스를 통해서 실험에 필요한 값을 입력한 후 시뮬레이션 프로그램을 실행한다. 시뮬레이션 결과는 가시화 시스템을 통해서 사용자에게 보여지고, 사용자는 결과 영상을 보면서 파라메터를 수정하고 다시 시뮬레이션을 실행시키는 식의 작업이 계속되었던 것이다. 문제는 시뮬레이션의 진행 중에는 사용자가 그 시뮬레이션을 제어하는 것이 불가능하다는 점이다. 또한 각 단계별 작업이 필요로 하는 시간이 매우 큰데다가 각 단계의 작업은 이전 단계의 작업이 반드시 끝나야 실행이 가능하기 때문에 그에 따른 시간 및 비용의 낭비도 심각한 문제였다.

이러한 문제점을 해결하기 위해서 computational steering system의 개념이 소개되었다. 이 시스템의 가장 큰 특징은 위에 나열한 각각의 과정을 동시에 (concurrently) 실행할 수 있도록 지원한다는 점이다. 사용자는 가시화 장비를 통해서 시뮬레이션이 진행되는 과정을 지켜볼 수 있으며, 사용자가 원하는 때에 필요한 파라메터를 변경시키면 시뮬레이션 프로그램은 변경된 값을 이용해서 계속 실행을 하고, 그 결과가 사용자에게 즉각적으로 보여지게 된다. 이러한 시스템을 이용하면 기존의 컴퓨터 시뮬레이션에서 볼 수 있었던 불필요한 시간 낭비를 줄일 수 있기 때문에 더 효율적인 실험을 수행할 수 있게 되고, 더 효과적으로 실험을 제어할 수 있게 된다.

일반적인 computational steering system은 그림 2와 같이 구성되어 있다.[2]

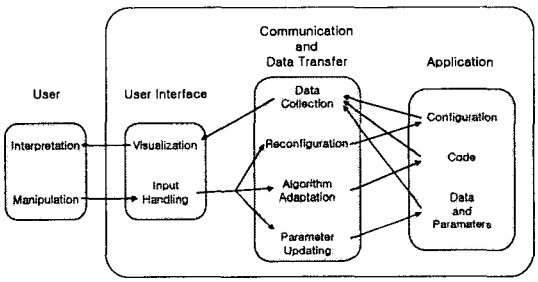

그림 2 Computational Steering System의 구성

· 사용자 인터페이스

사용자 인터페이스는 다시 가시화 시스템과 입력장치로 나누어서 생각할 수 있다. 가시화 시스템은 시뮬레이션 결과를 사용자에게 보여주기 위해 사용한다. 여기에는 고성능의 그래픽 전용 워크스테이션이 사용되며, 최근 하드웨어의 가격이 저렴해지면서 일반 PC가 이용되기도 한다. 입력장치는 사용자로 하여금 시뮬레이션 환경의 제어(파라메터 변경, 시

뮬레이션 진행 과정 통제 등)를 보다 손쉽게 할 수 있도록 지원해 주는 장치이다. 참고로, CAVE와 같은 고가의 몰입형 가상현실 시스템(immersive virtual reality system)은 가시화 시스템과 입력장치를 통합한 형태로 볼 수 있다.

· 통신 시스템

통신 시스템은 사용자 인터페이스와 시뮬레이션 프로그램 사이의 데이터 교환이 원활히 이루어질 수 있도록 지원해주는 시스템이다. Computational steering system은 사용자와의 상호작용(interaction)이 매우 빈번하게 일어나며, 대량의 데이터를 다룬다는 특징이 있다. 때문에 통신 시스템은 대용량의 데이터를 안정적으로 빠르게 주고받을 수 있도록 구성되어야 한다.

· 시뮬레이션 프로그램

실제로 시뮬레이션을 수행하는 부분이다. 앞에서도 설명했듯이 computational steering system은 사용자와의 상호작용(interaction)을 매우 중요시한다는 특징이 있기 때문에 프로그램도 그에 따라 개선되어야 한다. 또한 사용자가 C나 C++와(과) 같은 프로그래밍 언어를 이용해서 어플리케이션을 직접 작성하는 것도 가능하기는 하지만 사용자 입장에서는 보다 쉽게 코드를 작성할 수 있는 - visual programming과 같은 - 방법이 제공되어야 한다.

이밖에도 computational steering system을 구성하기 위해서 고려해야 될 사항들이 있다.

· 협업 환경의 지원

기존의 computational steering system은 한 순간에 단 한명의 사용자만 이용할 수 있는 시스템이 주류를 이루었다. 그러나 네트워크의 속도가 빨라지고 CAVE와 같은 가상현실 시스템이 발전하면서 최근에는 지역적으로 멀리 떨어진 다수의 사람들이 가상적으로 꾸며진 환경에 들어와서 동시에 작업을 할 수 있도록 지원하는 시스템의 개발이 활발하게 진행되고 있다.

· Grid의 지원

일반적으로 컴퓨터를 이용해서 대규모 시뮬레이션을 수행하는 데에는 많은 양의 컴퓨팅 자원을 필요로 하기 마련이다. 게다가 시간이 흐를수록 시뮬레이션의 규모가 커지기 때문에 단일 컴퓨터 시스템으로 그러한 요구를 충족시키는 것은 사실상 불가능하게 되었다. 때문에 다수의 컴퓨터와 대용량 저장장치등을 고속의 네트워크로 연결한 고성능 컴퓨팅 자원을 확보할 필요성이 갈수록 커지고 있다. 최근 여러 기관에서 연구를 하고 있는 그리드는 그러한 환경을 구축하기 위한 최적의 해결방법으로 떠오르고 있다.

4. Computational Steering System의 예

이 장에서는 실제로 구현되어서 운영하고 있는 computational steering system들을 소개한다.

4.1 SCIRun[3,7]

SCIRun은 Utah 대학의 SCI 그룹에서 제작한 시스템이다. 이 툴의 특징은 데이터 플로우(data-flow) 프로그래밍 모델을 지원한다는 점이다. 이를 이용하면 사용자는 소스 코드를 일일이 입력하지 않고 GUI를 통한 간단한 조작만으로 프로그램을 작성해서 실행할 수 있게 된다. SCIRun의 데이터 플로우 모델에서는 프로그램을 구성하는 각각의 모듈이 사각형으로 표현된다. 그리고 각 모듈을 연결하는 선은 한 모듈에서 데이터를 처리한 결과가 다른 모듈의 입력이 됨을 나타낸다. 초창기 SCIRun은 공유 메모리 구조를 갖는 컴퓨터에서만 실행이 되도록 제작되었으나 분산 메모리 구조를 갖는 컴퓨터에서도 실행될 수 있도록 개선되었다. 또한 Globus Toolkit과도 연동될 수 있도록 구성되어 있다.

4.2 VIPER[4]

VIPER는 VIsualization of massively Parallel simulation algorithms for Extended Research의 줄임말로, 독일의 Munich 대학에서 만들어졌다. 이 툴은 계산유체역학(Computational Fluid Dynamics, CFD) 분야의 문제 해결에 특화되어 있으며, 프로세스를 여러 대의 컴퓨터에 분산시켜서 실행할 수 있는 기능을 가지고 있다.

4.3. CAVEStudy[5, 9]

CAVEStudy는 CAVERNsoft, Globus Toolkit 등을 기반으로 작성되었으며, 가상현실 환경 내에서의 프로그램 실행 제어를 위한 툴킷이다. CAVEStudy의 특징은 기존의 시뮬레이션 프로그램을 별도로 수정하지 않고 곧바로 적용시킬 수 있도록 구성되어

있다는 점이다. CAVEStudy는 크게 코드 생성기(code generator)와 가상현실 프레임워크(VR framework)로 구성되어 있다. 코드 생성기는 사용자가 기술한 description file을 이용해서 시뮬레이션의 실행 및 제어와 관련된 C++ 코드를 생성한다. 가상현실 프레임워크는 시뮬레이션 결과의 가시화 및 시뮬레이션의 수행 제어에 사용된다.

4.4 CUMULVS[6,8]

CUMULVS는 ORNL에서 개발되었다. 이 시스템의 특징은 fault-tolerance 기능을 제공한다는 점이다. CUMULVS는 어플리케이션을 실행할 때 일정 시간 간격을 두고 현재의 상태를 저장해 두었다가 문제가 발생해서 어플리케이션의 실행이 중단되면 가장 최근에 저장해둔 상태를 다시 불러와서 실행을 계속 하도록 구성되어 있다.

종 류	특 징
SCIRun	• Visual programming model • Tcl/Tk interface
CAVEStudy	• Collaborative Environment • CAVE Interface
VIPER	• Massively parallel • Proprietary GUI
CUMULVS	• Fault-tolerance • AVS interface

표 1 Computational Steering System의 비교

5. N* Grid를 위한 Computational Steering System

KISTI 슈퍼컴퓨팅센터에서는 국가 그리드 기반인 N* Grid를 위한 computational steering system에 대한 연구를 진행하고 있다. 이 시스템은 앞에서 설명한 주요 computational steering system의 특징을 최대한 수용해서 다음과 같이 구현할 예정이다.

- 사용자 인터페이스와 가시화 시스템은 CAVE로 구현한다.
- Visual programming tool을 제공해서 사용자가 프로그램을 작성할 때 일일이 코드를 작성하지 않아도 시뮬레이션을 수행할 수 있도록 한다.
- 시뮬레이션 코드는 N* Grid를 충분히 활용하도록 구성한다.
- 지역적으로 떨어진 여러 명의 사용자가 가상 환경에 들어와서 작업할 수 있도록 지원한다.

6. 결론

본 논문에서는 computational steering system의 개념을 설명하고 그 특징에 대해 살펴보았으며, 실제로 사용되고 있는 몇몇 computational steering system에 대해 설명하였다.

현재 KISTI 슈퍼컴퓨팅센터에서는 구가 그리드 기반인 N* Grid에서 운용할 computational steering system에 대한 연구를 진행하고 있다.

참고문헌

[1] 그리드 연구개발 동향

[2] Jurriaan D. Mulder, Jarke J. van Wijk, Robert van Liere, "A Survey of Computational Steering Environments", Future Generation Computer Systems, Vol. 15, No. 1, pp.119-129, 1999.

[3] Steven G. Parker, David M. Weinstein, Christopher R. Johnson, "The SCIRun Computational Steering Software System", Modern Software Tools in Scientific Computing, 1997.

[4] Gregor von Laszewski, et al, "Designing Grid-based Problem Solving Environments and Portals", 34th Annual Hawaii International Conference on System Sciences, Vol. 9, Januray, 2001.

[5] Luc Renambot, et al, "CAVEStudy: an Infrastructure for Computational Steering in Virtual Reality Environments", Proceedings of the 9th IEEE International Symposium on High Performance Distributed Computing, pp 57-61, 2000.

[6] G. A. Geist, J. A. Kohl, P. M. Papadopoulos, "CUMULVS: Providing Fault-Tolerance, Visualization and Steering of Parallel Applications', International Journal of High Performance Computing Applications, Vol. 11, No. 3, pp. 224-236, 1996.

[7] http://www.sci.utah.edu

[8] http://www.csm.ornl.gov/cs/cumulvs.html

[9] http://www.cs.vu.nl/~renambot/vr/

[10] http://www.globus.org