서론

과학적 계산은 고급 컴퓨팅 능력을 활용하여 복잡한 문제를 이해하고 해결하는 과학 분야입니다. 물리학, 화학, 생물학, 공학 등 다양한 과학 분야에서 수치 계산, 시뮬레이션, 데이터 분석을 수행할 수 있는 수학 모델, 알고리즘, 소프트웨어의 개발과 응용을 포함합니다. 과학적 계산은 이론과 실험과 함께 과학의 세 번째 방식으로 볼 수 있으며, 과학적 발견 과정을 보완하고 향상시킬 수 있습니다.

Scientific Computing은 고성능 수치 연산 능력을 활용하여, 실제의 복잡한 현상을 이해하고 해결하는 과학의 한분야를 뜻한다. 여기에는 물리학, 화학, 생물학, 공학 등의 다양한 과학 분야에서 수치 해석, 시뮬레이션, 데이터 분석을 수행하기 위한 수학적 모델, 알고리즘, 소프트웨어의 개발과 응용이 포함된다. Scientific Computing은 이론 및 실험과 함께 과학 분야의 세번째 도구로 활용되고 있으며, 이를 통하여 과학적 발견 과정을 보완하고 향상시킬 수 있어, 고성능 컴퓨터의 발전과 더불어 널리 활용되고 있다. 최근에는 AI/ML의 발전과 함께 경제학, 심리학 등 사회과학 연구에도 Scientific Computing의 활용이 증가하고 있다.

Scientific Computing을 활용하기 위해서는 수학, 전산학, 프로그래밍 언어, 소프트웨어 공학, 하드웨어 아키텍처와 같은 다양한 분야의 배경 지식과 기술이 요구된다. 이러한 기본 지식 외에도, 슈퍼컴퓨터, HPC 클러스터, 및 클라우드 컴퓨팅과 같은 고성능 연산 자원이 필요하다. 이와 함께 Scientific Computing을 활용하고자 하는 분야에 대한 지식 (Domain Knowledge)가 필수적이다.

이러한 이유로, Sdomain knowledge와 함께 scientific computing의 배경이 되는 분야별 수학적 지식과 알고리즘, 이를 구현하기 위한 프로그래밍 언어에 대한 적절한 교육과정이 필수적이며 이를 수행하기 위한 하드웨어 아키텍쳐 등 IT 지식과 기술에 대한 단계적인 교육 과정이 필요하다.

이 글에서는 scientific computing에 널리 활용되고 있는 프로그래밍 언어, 수치 라이브러리와 이에 대한 교육 방안에 대해서 살펴보고, 슈커컴퓨터, HPC 클러스터 및 클라우드 컴퓨팅 등 관련 IT 지식과 기술에 대한 교육과 학습 방안에 대해서 알아보고자 한다.

Scienitific Computing을 위한 대표적인 프로그래밍 언어

Scientific computing 어플리케이션이나 이 어플리케이션이 수행되는 하드웨어 및 프로그래머의 선호도에 따라 scientific computing에 적합한 프로그래밍 언어는 달라질 수 있다. 그럼에도 불구하고, 전통적으로 scientific computing에 널리 사용되어 왔고, 현재도 사용되는 대표적인 프로그래밍 언어로는 다음을 들 수 있다.

1) 포트란 (Fortran)

포트란은 과학 연산을 위한 범용 언어로 가장 빠른 성능을 보이는 10가지 프로그래밍 언어

(https://www.geeksforgeeks.org/top-10-fastest-programming-languages/)에 포함되는 등, 큰 배열 데이터를 다르는데 있어서 다른 범용 프로그래밍 언어보다 빠르고 효율적이다. 이러한 이유로 수치 연산 관련 프로그래밍에 사용되기 위해 개발되었으며, 오랜 역사와 최적화를 거쳐 많은 수치 연산 관련 라이브러리, 코드와 호환되는 scientific computing 분야의 대표적인 프로그래밍 언어이다.

2) C/C++

C와 C+은 프로그램의 메모리 관리와 하드웨어와의 상호작용을 세밀하게 제어할 수 있는 로우레벨의 프로그래밍 언어이다. C와 C+은 병렬 및 분산 컴퓨팅을 가능하게 하는 많은 컴파일러, 라이브러리, 프레임워크를 널리 지원하면서 운영체제, 드라이버, 커널과 같은 시스템 프로그래밍이나 속도와 유연성이 필요한 고성능 응용 프로그램에 널리 사용되는 언어이다.

3) 파이썬 (Python)

파이썬은 읽고 쓰고 배우기 쉬운 고수준 언어로 데이터 분석, 머신러닝, 시각화, 웹 개발 등 다양한 기능을 제공하는 크고 풍부한 라이브러리와 모듈을 제공하고 있다. 또한 실행하기 전에 컴파일할 필요가 없는 해석형 언어(interpreted language)로 더 휴대성이 높고 사용이 편리하다. 그러나 포트란이나 C/C++과 같은 컴파일 언어보다 느리기 때문에 성능이 중요한 작업보다는 다른 언어나 라이브러리를 호출하는 스크립팅 언어나 인터페이스 개발용 언어로 많이 사용된다.

이상이 scientific computing을 위해 널리 활용되고 인기 있는 프로그래밍 언어 중 일부이다. 하지만 각 분야의 요구 사항과 개발자의 취향에 따라 다양한 프로그래밍 언어들이 활용되고 있다. 예를 들어 Julia, R, MATLAB 등이 다양한 분야의 scientific computing에 활용되고 있다. 프로그래밍 언어의 선택은 해결하고자 하는 문제의 영역, 사용 가능한 자원, 프로그래머의 숙련도, 속도와 신뢰성 사이에서 원하는 트레이드 오프 등 여러 요인에 따라 달라질 수 있다. 따라서 scientific computing을 위한 최고의 프로그래밍 언어를 특정할 수는 없고 상황에 따라 사용할 수 있는 다양한 프로그래밍 언어가 존재한다.

Scientific computing을 위한 프로그래밍 언어 순위

수치 해석, 행렬 해석자(matrix solver), 데이터 분석 등 풀고자 하는 다양한 scientific computing 문제에 따라 적합한 프로그래밍 언어가 존재하는데, 이러한 언어들의 순위는 기준, 출처, 순위를 측정하는 시기에 따라 달라질 수 있다. scientific computing을 위한 프로그래밍 언어에 대한 대표적인 순위는 다음의 결과에서 살펴볼 수 있다.

기술과 공학 주제를 다루는 잡지인 IEEE Spectrum의 2022년 조사(https://spectrum.ieee.org/top-programming-languages-2022) 에 따르면 scientific computing을 위한 상위 10개의 프로그래밍 언어는 파이썬, C, C++, 포트란, R, Julia, MATLAB, Java, JavaScript, SQL을 들 수 있다. 이 순위는 웹 검색, 소셜 미디어 언급, 채용 공고, 오픈소스 프로젝트 등 다양한 출처에서 얻은 여러 지표를 결합한 것으로 이 순위는 다른 지표의 가중치를 조정하고 다른 플랫폼과 도메인으로 필터링할 경우 바뀔 수 있다. 또한 2023년 조사(https://spectrum.ieee.org/the-top-programming-languages-2023)에 따르면, 여전히 가장 인기 있는 프로그래밍 언어는 파이썬이다. 파이썬은 scientific computing 분야뿐만 아니라, AI와 같은 분야에서도 강력하고 광범위한 라이브러리로 인해 널리 사용되고 있다. 또한 저렴한 CPU 에서도 파이썬을 사용할 수 있는 충분한 컴퓨팅 파워가 제공되고 있어 임베디드 개발에서도 유용하게 사용되고 있다. 다음으로 인기 있는 언어는 SQL, C, Java, C++, R 등이다.

사용자들의 리뷰와 투표를 바탕으로 다양한 제품과 서비스에 대한 추천을 제공하는 웹사이트인 Slant (https://www.slant.co/topics/16647/~languages-for-scientific-computation) 에서는 2023년 scientific computing을 위한 최고의 언어로 Julia, Python, C, Fortran, R, Nim, Wolfram Mathematica, C++, MATLAB을 추천하고 있다. 이 순위는 사용자들의 의견과 경험, 그리고 각 언어의 장단점을 고려한 것이다. 이와 함께 각 언어를 배우기 위한 튜토리얼과 자료들에 대한 링크도 제공하고 있다.

Flatiron School이라는 온라인 교육 플랫폼에 따르면, 각 언어의 인기도, 수요, 다용도성, 기능성 등을 고려한 2021년 데이터 과학 분야에서 사용되는 상위 10개의 프로그래밍 언어 (https://flatironschool.com/blog/data-science-programming-languages/)는 파이썬, 자바스크립트, 자바, R, C/C++, SQL, MATLAB, 스칼라, SAS, Ruby 이다.

어린이와 성인을 위한 로봇 교육과 제품을 제공하는 Robotical.io (https://robotical.io/blog/the-10-best-data-science-programming-languages-to-learn/) 라는 웹사이트에 따르면, 2022년 현재 배워야할 데이터 과학 프로그래밍 언어 상위 10개로는 파이썬, 자바스크립트, 자바, SQL, R, Scala, Julia, C/C++, MATLAB, SAS를 들고 있다. 이 순위는 각 언어의 학습 용이성, 유용성, 성능, 데이터 과학 분야에서의 사용 가능성 등을 고려한 것이다.

마지막으로, edX((https://www.edx.org/resources/9-top-programming-languages-for-data-science) 라는 비영리 온라인 학습 플랫폼에 따르면, 데이터 과학 분야에서 사용되는 상위 9개의 프로그래밍 언어는 파이썬, R, SQL, 자바, C/C++, MATLAB, 스칼라, 줄리아, SAS이다. 이 순위는 각 언어의 특징, 장점, 단점, 사용 사례 등을 고려한 것이다.

이상의 다양한 조사에서 볼 수 있듯이 scientific computing을 위한 최고의 프로그래밍 언어에 대한 명확한 답이나 순위는 알 수 없다. 각각의 프로그래밍 언어는 문제 도메인, 사용 가능한 자원, 프로그래머의 기술 수준 등에 따라 각각의 장단점이 있다. 따라서 scientific computing을 위한 프로그래밍 언어를 선택할 때에는 사용자의 목표와 선호도를 고려하는 것이 중요하다고 할 수 있다. 그러나, 다양한 조사와 전통적인 계산 과학 분야, 최근의 데이터 과학, 인공지능 등, 사용자의 선호도 등을 고려할 때, Python, C/C++, MATLAB, R 등이 scientific computing을 위한 대표적인 프로그래밍 언어로 볼 수 있다. 이와 함께 Julia도 여러 곳에서 scientific computing을 위한 프로그래밍 언어로 많이 추천되고 있음을 알 수 있다.

IEEE Spectrum 2022	IEEE Spectrum 2023	Slant 2023	Flatiron School 2021	Robotical.io 2022	edX
Python C/C++ Fortran R Julia MATLAB Java JavaScript SQL	Python SQL C/C++ Java R	Julia Python C/C++ Fortran R Nim Mathematica MATLAB	Python JavaScript Java R C/C++ SQL MATLAB Scala SAS Ruby	Python JavaScript Java SQL R Scala Julia C/C++ MATLAB SAS	Python R SQL Java C/C++ MATLAB Scala Julia SAS

Scientific computing을 위한 수치 라이브러리

Scientific computing에 널리 활용되는 대표적인 수치 라이브러리와 이를 지원하는 언어는 다음과 같다.

1) GNU Scientific Library (https://www.gnu.org/software/gsl/)

GNU 과학 라이브러리(GSL)는 C와 C++ 프로그래머를 위한 수치 라이브러리로 GNU Public 라이선스로 배포된다. 이 라이브러리는 난수 생성기, 특수 함수, 최소 제곱 적합 등과 같은 다양한 수학적 루틴을 제공하며, 총 1000개 이상의 함수와 광범위한 테스트 스위트가 포함되어 있다.

2) BLAS (Basic Linear Algebra Subprograms, https://www.netlib.org/blas/)

1979년 포트란 라이브러리로 처음 개발된 기본 선형대수 서브프로그램인 BLAS는 벡터 덧셈, 스

칼라 곱셈, 내적, 선형 조합, 행렬 곱셈과 같은 일반적인 선형대수 연산을 수행하기 위한 저수준 루틴의 집합으로C ("CBLAS 인터페이스")와 포트란 ("BLAS 인터페이스") 모두에 대한 바인딩을 제공한다. BLAS는 특정 하드웨에 최적화된 라이브러리(Intel MKL, NVIDIA CuBLAS 등)를 제공하므로 이를 이용하면 상당한 성능 향상을 얻을 수 있다. LAPACK, LINPACK, Armadillo, GNU Octave, Mathematica, MATLAB, NumPy, R, Julia 및 Lisp-Stat을 포함한 많은 수치 소프트웨어 응용 프로그램은 BLAS 호환 라이브러리를 사용하여 선형대수 계산을 수행한다.

3) LAPACK (Linear Algebra PACKage, https://www.netlib.org/lapack/)

LAPACK은 선형대수 연산, 예를 들어 방정식 시스템, 고유값 문제, 특이값 문제 등을 수행하기 위한 소프트웨어 라이브러리로 이것은 포트란으로 작성되었지만 C와 C++에 대한 인터페이스도 제공한다. 그리고 MATLAB과 같은 소프트웨어에서의 선형 대수 함수와 행렬 연산은 LAPACK을 기반으로 하고 있으며, 그 루틴의 성능과 정확도를 계속해서 활용하고 있다. BLAS와 마찬가지로 대부분의 scientific computing에 많이 사용되는 프로그래밍 언어에서는 LAPACK을 지원하고 있다.

4) NumPy (https://numpy.org/)

NumPy는 파이썬 프로그래밍 언어를 위한 수치 라이브러리로, 큰 다차원 배열과 행렬을 지원하고, 이러한 배열에 작용하는 고수준 수학 함수의 큰 모음을 제공한다.

5) SciPy (https://docs.scipy.org/doc/scipy/)

SciPy는 Python 기반의 수학, 과학, 공학을 위한 오픈 소스 소프트웨어로 통계, 최적화, 적분, 선형대수, 푸리에 변환, 신호 및 이미지 처리, ODE 해석자 등 다양한 종류의 수치 계산을 위한 많은 모듈과 함수를 제공한다. SciPy는 효율적인 배열 조작을 위한 라이브러리인 NumPy를 기반으로 구현되어 있다. 이와 함께 희소 행렬과 k-차원 트리와 같은 전문적인 데이터 구조와 과학 계산을 위한 기본 알고리즘이 NumPy를 기반으로 확장되어 있다. BLAS와 LAPACK과 같은 수치 라이브러리도 SciPy를 통해 지원한다.

6) Pandas (https://pandas.pydata.org/)

Pandas는 파이썬 프로그래밍 언어를 기반으로 하는 빠르고 강력하고 유연하고 사용하기 쉬운 오픈 소스 데이터 분석 및 조작 도구입니다.

Scientific computing을 위해 특화된 소프트웨어 및 언어

수치라이브러리 외에도, 수치연산을 위해 개발된 다음의 소프트웨어들이 scientific computing 분야에서 널리 활용되고 있다.

1) MATLAB

MATLAB은 수치 계산을 수행하기 위해 널리 사용되는 상용 소프트웨어이다. MATLAB은 자체 프로그래밍 언어를 제공하는데 이는 GNU Octave라는 오픈 소스 소프트웨어와 대부분 호환된다. MATLAB은 C, C++, Java, Python 등 다른 언어에서 작성된 함수를 호출하는 것도 지원하며 행렬 조작과 해석을 위한 많은 내장 함수와 툴박스를 제공한다.

2) R

R은 통계 계산과 그래픽을 위한 프로그래밍 언어와 환경을 제공하는데, 벡터, 행렬, 배열, 데이터 프레임 등과 같은 데이터 구조를 조작하기 위한 많은 기능을 가지고 있다. 또한 Matrix, lme4, MASS 등과 같은 수치 분석을 수행하기 위한 많은 패키지를 포함하고 있으며 선형 시스템을 풀거나, 고유값과 고유벡터를 찾거나, 행렬 분해를 계산하는 등의 함수를 제공한다.

3) Julia

Julia는 MATLAB과 표면적으로 유사한 고수준 동적 언어로 고성능 컴퓨팅을 위해 설계되었으며 다중 디스패치, 병렬성, 메타 프로그래밍을 지원한다. 또한 데이터 분석, 기계 학습, 최적화 등 다양한 분야를 위한 패키지를 포함하고 있으며, 선형 대수를 위한 표준 라이브러리를 제공하며 선형 시스템을 풀거나, 고유값과 고유벡터를 찾거나, 행렬 인수분해를 계산하는 등의 함수를 제공한다.

MPI 프로그래밍을 지원하는 언어

MPI 프로그래밍은 C, C++, Fortran, Java, Python, R, MATLAB 등 다양한 언어에서 지원된다. 이 중 일부 언어는 MPI에 대한 네이티브 바인딩을 가지고 있고, 다른 언어는 외부 라이브러리나 패키지를 사용하여 MPI 기능에 접근한다. 다음은 다른 언어들이 MPI를 사용하는 예시이다.

1) C

C 언어에서 MPI를 사용하려면 mpi.h 헤더 파일을 포함해야한다. MPI 프로그램은 mpicc 컴파일러로 컴파일하고, mpirun 명령어로 실행할 수 있다. MPI 함수는 MPI_ 접두사로 시작한다. 예를 들어, MPI_Init, MPI_Comm_size, MPI_Send, MPI_Recv 등이 있다.

2) C++

C++ 언어에서 MPI를 사용하려면 mpi.h 헤더 파일을 포함해야 하며 C 언어와 마찬가지로 mpiCC 컴파일러와 mpirun 명령어를 사용한다. C++에서는 MPI 네임스페이스를 사용하여 MPI 함수에 접근할 수 있다. 예를 들어, 에 접근할 수 있습니다. 예를 들어, MPI::Init, MPI::COMM_WORLD.Get_size, MPI::COMM_WORLD.Send, MPI::COMM_WORLD.Recv 등이 있다.

3) Fortran

Fortran 언어에서 MPI를 사용하려면 mpif.h 모듈을 포함해야 한다. MPI 프로그램은 mpif90 컴파일러로 컴파일하고, mpirun 명령어로 실행한다. MPI 함수는 대문자로 쓰이고, 인자는 명시적으로 선언되어야 한다. 예를 들어, CALL MPI_INIT, CALL MPI_COMM_SIZE, CALL MPI_SEND, CALL MPI_RECV 등이 있다.

4) Java

Java 언어에서 MPI를 사용하려면 mpi.jar 파일을 클래스 패스에 추가해야 한다. MPI 프로그램은 일반적인 Java 프로그램과 같이 컴파일하고, mpirun 명령어로 실행한다. Java에서는 mpi 패키지를 임포트하여 MPI 객체와 메소드에 접근할 수 있다. 예를 들어, MPI.Init, MPI.COMM_WORLD.Size, MPI.COMM_WORLD.Send, MPI.COMM_WORLD.Recv 등이 있다.

5) Python

Python 언어에서 MPI를 사용하려면 mpi4py 패키지를 설치해야 한다. Python 스크립트는 일반적인 Python 스크립트와 같이 작성하고, mpirun 명령어로 실행할 수 있다. Python에서는 mpi4py 모듈을 임포트하여 MPI 객체와 메소드에 접근할 수 있다. 예를 들어, mpi4py.MPI.Init, mpi4py.MPI.COMM_WORLD.Get_size, mpi4py.MPI.COMM_WORLD.Send, mpi4py.MPI.COMM_WORLD.Recv 등이 있다.

6) R

R 언어에서 MPI를 사용하려면 Rmpi 패키지를 설치해야 한다. R 스크립트는 일반적인 R 스크립트와 같이 작성하고, Rmpi 패키지의 함수를 사용하여 MPI 프로그램을 시작하고 종료할 수 있다. R에서는 Rmpi 패키지의 함수를 사용하여 MPI 기능에 접근할 수 있다. 예를 들어, mpi.init, mpi.comm.size, mpi.send, mpi.recv 등이 있다.

7) MATLAB

MATLAB 언어에서 MPI를 사용하려면 Parallel Computing Toolbox를 설치해야 한다. MATLAB 스크립트는 일반적인 MATLAB 스크립트와 같이 작성하고, MATLAB의 명령창에서 mpirun 명령어로 실행할 수 있다. MATLAB에서는 mpi 패키지를 사용하여 MPI 객체와 메소드에 접근할 수 있다. 예를 들어, mpi.init, mpi.comm.size, mpi.comm.send, mpi.comm.recv 등이 있다.

8) Julia

Julia는 mpi4py에 영감을 받은 MPI에 대한 Julia 인터페이스를 제공하는 MPI.jl이라는 패키지가 있다. Pkg 매니저를 사용하여 MPI.jl을 설치하고 여러 프로세스 간에 통신하는 병렬 프로그램을 작성할 수 있다. 또한 표준 라이브러리의 Distributed 모듈을 사용하여 분산 메모리 병렬 컴퓨팅을 Julia에서 구현할 수 있다.

이상에서 알 수 있듯이 앞에서 논한 scientific computing에 널리 활용되고 있는 프로그래밍 언어들인 Python, C/C++, Fortran, MATLAB, R, Julia 등은 대부분의 수치 라이브러리뿐만 아니라 병렬 프로그래밍의 표준이라고 할 수 있는 MPI도 지원하고 있다.

Scientific computing 프로그래밍 언어 교재

이미 C/C++과 함께 Python, MATLAB, R을 scientific computing에 활용하기 위한 다음과 같은 교재가 출판되어 있고, 이를 활용하여 도메인 지식과 더불어 scientific computing을 교육하고 학습함으로써 엑사스케일 슈퍼컴퓨팅시대를 대비하는 것이 가능할 것으로 생각된다.

1) Guide to Scientific Computing in C++ (https://link.springer.com/book/10.1007/978-3-319-73132-2)

2) Scientific Computing with Python (https://www.amazon.com/Scientific-Computing-Python-High-performance-scientific/dp/1838822321)

3) Scientific Computing with MATLAB(https://link.springer.com/book/10.1007/978-3-642-59339-0)

4) Mastering Scientific Computing with R (https://www.amazon.com/Mastering-Scientific-Computing-Paul-Gerrard/dp/1783555254)

5) Python for the Scientific Computing https://aaltoscicomp.github.io/python-for-scicomp/

결론

앞에서 논한 바와 같이 다양한 scientific computing을 위한 프로그래밍 언어 순위, 대표적인 수치 라이브러리 지원, 병렬 프로그램의 표준이라고 할 수 있는 MPI 지원 등을 고려할 때, scientific computing을 위한 프로그래밍 언어로는 전통적으로 널리 사용되어 온, Fortran과 C/C++ 이외에도 Python, Java, MATLAB, R 등 scientific computing 뿐만 아니라 다양한 IT 분야, AI/ML 분야에서 널리 사용되고 있는 프로그래밍 언어들이 scientific computing에 활용된다고 볼 수 있다. 이에 따라, 이런 언어를 기반으로 scientific computing에 대한 교육과정과 학습이 필요하다고 본다.

특히, 파이썬, JAVA, R 와 같은 언어는 scientific computing 뿐만 아니라, 웹개발 등의 IT 분야에서도 대표적으로 활용되는 프로그래밍 언어이며 최근 AI/ML, 데이터 분석 분야의 활용에 따라 과학기술계 외에 인문사회 계열, 경제학 등에서도 널리 이용되고 있고 교육되는 프로그래밍 언어이다. 따라서 Scientific computing을 위한 교육과정으로는 Python, R, JAVA 등을 기반으로 도메인 지식과 수치 라이브러리를 교육하고 이를 바탕으로 C/C++ 등 로우레벨의 언어와 함께 하드웨어, 인프라, 클라우드 컴퓨팅 등의 심화 교육과정을 개발하는 것이 필요하다고 본다.