한국농림기상학회지, 제21권 제1호(2019) (pISSN 1229-5671, eISSN 2288-1859)
Korean Journal of Agricultural and Forest Meteorology, Vol. 21, No. 1, (2019), pp. 65~73
DOI: 10.5532/KJAFM.2019.21.1.65

단 보

운영 체제와 컴파일러에 따른 Geospatial Data Abstraction Library의 Hierarchical Data Format 형식 원격 탐사 자료 추출 속도 비교

유병현[1], 김광수[1,2*], 이지혜[3]
[1]서울대학교 식물생산과학부, [2]서울대학교 농업생명과학연구원, [3]국가농림기상센터
(2019년 2월 8일 접수; 2019년 3월 7일 수정; 2019년 3월 13일 수락)

Comparison of the wall clock time for extracting remote sensing data in Hierarchical Data Format using Geospatial Data Abstraction Library by operating system and compiler

Byoung Hyun Yoo[1], Kwang Soo Kim[1,2*] and Jihye Lee[3]
[1]Department of Plant Science, Seoul National University, Seoul, Korea
[2]Research Institute of Agriculture and Life Science, Seoul National University, Seoul, Korea
[3]National Center for Agro-Meteorology, Seoul, Korea
(Received February 8, 2019; Revised March 7, 2019; Accepted March 13, 2019)

ABSTRACT

The MODIS (Moderate Resolution Imaging Spectroradiometer) data in Hierarchical Data Format (HDF) have been processed using the Geospatial Data Abstraction Library (GDAL). Because of a relatively large data size, it would be preferable to build and install the data analysis tool with greater computing performance, which would differ by operating system and the form of distribution, e.g., source code or binary package. The objective of this study was to examine the performance of the GDAL for processing the HDF files, which would guide construction of a computer system for remote sensing data analysis. The differences in execution time were compared between environments under which the GDAL was installed. The wall clock time was measured after extracting data for each variable in the MODIS data file using a tool built lining against GDAL under a combination of operating systems (Ubuntu and openSUSE), compilers (GNU and Intel), and distribution forms. The MOD07 product, which contains atmosphere data, were processed for eight 2-D variables and two 3-D variables. The GDAL compiled with Intel compiler under Ubuntu had the shortest computation time. For openSUSE, the GDAL compiled using GNU and intel compilers had greater performance for 2-D and 3-D variables, respectively. It was found that the wall clock time was considerably long for the GDAL complied with "--with-hdf4=no" configuration option or RPM package manager under openSUSE. These results indicated that the choice of the environments under which the GDAL is installed, e.g., operation system or compiler, would have a considerable impact on the performance of a system for processing remote sensing data. Application of parallel computing approaches would improve the performance of the data processing for the HDF files, which merits further evaluation of these computational methods.

***Key words*: MODIS, GDAL, HDF, Compiler, Operating system**

* Corresponding Author : Kwang Soo Kim
(luxkwang@snu.ac.kr)

I. 서 론

최근 기후변화와 이에 따른 작황의 불확실성이 증대됨에 따라 지역 규모에서 작물 생산성을 추정하기 위한 시스템들이 개발되고 있다(Lobell *et al*., 2003; Prasad *et al*., 2006). 특히, 광역 규모의 작물 생육 모니터링을 위해 준실시간으로 제공되는 인공위성 자료가 활용되고 있다(Turner *et al*., 2006; Jobbagy *et al*., 2002; Doraiswamy *et al*., 2004). 해외 주요 곡물 생산국에서는 위성영상 기반의 작물 모니터링 시스템을 활용하여 작황 정보를 정기적으로 생산 및 제공하고 있다. 예를 들어, 미국, 유럽연합 및 중국은 각각National Agricultural Statistics Service, Monitoring Agriculture ResourceS, CropWatch 등의 작물 모니터링 시스템을 구축하여 작물 생산성 자료를 생산하고 있다.

국내에서도 위성 영상자료 기반의 국외 작물 생산 모니터링 체계가 개발되고 있다(Lee *et al*., 2011; Lee *et al*., 2017). 이러한 작황 모니터링에 필수적으로 사용되는 입력자료를 얻기 위해 MODIS (Moderate Resolution Imaging Spectroradiometer), Landsat, Sentinel 등 다양한 위성의 영상 자료들이 사용되고 있다. 예를 들어, Ban *et al*.(2016)은 MODIS 자료를 활용하여 중국과 미국의 일리노이 주와 같이 대규모 재배가 이루어지는 국외 지역에 대한 옥수수 작황을 추정하였다. 또한, Hong *et al*.(2015)은 자료 수집이 어려운 북한 지역에 대해 Landsat자료를 활용하여 작황을 추정하는 시스템을 개발하였다.

인공위성 영상자료 중 MODIS 영상자료는 다양한 농업 생태계 정보 추정에 활용될 수 있는 정보를 제공하고 있다(Vancutsem *et al*., 2010). MODIS 영상자료에는 온도와 습도 등 작물 생육과 연관된 대기 자료인 MOD07 자료와 작물의 생육을 간접적으로 파악할 수 있는 지표 반사율 자료인 MOD09 자료 등이 포함되어 있다. 또한, 이러한 기초 자료를 가공하여 얻어지는 EVI, NDVI 및 LAI 자료 등 작물의 생육 추정에 활용될 수 있는 응용자료들도 제공되고 있다. MODIS 자료들은 다양한 자료들을 하나의 파일에 저장하여 제공할 경우, 여러 종류의 격자형 데이터를 하나의 파일에서 관리할 수 있는 HDF 파일 형식이 사용된다(Masuoka *et al*., 1998). 예를 들어, MOD07 Level 2 자료의 경우, 하나의 파일에 격자별 위도, 경도 및 온도 등의 자료가 함께 저장되어 있다.

HDF 파일 형식의 인공위성 자료를 활용하기 위해 격자자료 처리를 위한 전문 소프트웨어들이 사용된다. 예를 들어, 상용 소프트웨어인 ENVI와 이에 포함되어 있는 IDL은 HDF 자료를 처리하기 위해 사용될 수 있다. 그러나, ENVI의 경우 상용 소프트웨어의 라이센스를 유지하기 위해 상당한 비용이 소요된다. 한편, 오픈소스 형태로 제공되는 격자 자료 처리용 API나 프로그래밍 언어가 HDF 파일에 포함되어 있는 영상자료 처리를 위해 사용될 수 있다. 예를 들어, R이나 python등의 오픈소스 프로그램을 사용할 경우, HDF 자료 처리를 위한 비용을 절감할 수 있다.

HDF 파일을 처리하는 오픈소스 소프트웨어들은 주로 격자 자료 처리를 지원하는 GDAL 라이브러리에 의존한다(Busetto and Ranghetti, 2016). 이러한 공용 라이브러리의 경우, 사용자가 다양한 방식으로 설치할 수 있다. Linux 사용자의 경우 바이너리 파일로 배포되는 RPM package manager (RPM)나 Debian package (deb)의 형태로 GDAL 라이브러리가 설치된다. 패키지 형태의 경우, 사용자가 간단한 명령어나 GUI 환경에서 손쉽게 설치되기 때문에 시스템 구축에 편리하다. 사용자가 GDAL 라이브러리의 소스코드들을 컴파일하여 설치할 수도 있다. 소스코드를 컴파일하여 설치할 경우, 운영체제에 따라 사용자가 직접 여러가지 선택 사항들을 지정하여야 하기 때문에 일반적인 사용자에게는 까다로울 수 있다. 반면, 사용자의 하드웨어와 소프트웨어에 최적화된 라이브러리들을 구축할 수 있는 장점이 있다.

본 연구에서는 GDAL 라이브러리 구축 방식에 따른 HDF 자료 추출 속도를 비교해 보고자 하였다. 작황 모니터링 시스템의 경우 대규모 영상자료 처리가 요구되므로, 영상 입력자료의 효율적인 처리가 요구된다. 이 때, 위성 영상자료를 처리하기 위한 라이브러리의 설치 형식에 따라 처리속도의 차이가 발생할 수 있다(Guntheroth, 2016). 따라서, 위성영상자료 처리 도구의 설치 조건에 따른 성능 비교를 통해, 농업 분야에서 활용될 수 있는 최적의 인공위성 영상자료 처리 시스템 설계 및 구축을 지원할 수 있을 것이다.

II. 재료 및 방법

2.1. 위성영상자료

HDF 형식의 영상자료 추출시간을 측정하기 위해 Aqua 위성으로부터 생산된 2017년도 7월 19일 05시 55분 MODIS 대기 자료를 사용하였다. 이 자료에는

Table 1. The metadata of variables contained in the MOD07 product for measurement of the processing time using the GDAL

Variable Name	Size
Latitude Longitude Solar_Zenith Cloud_Mask Surface_Pressure Surface_Elevation Total_Ozone Water_Vapor	406x270
Retrieved_Temperature_Profile Retrieved_Moisture_Profile	406x270x20

북아메리카 지역의 북부 일부 지역이 포함되어 있으며 약 500m의 공간해상도를 가지고 있다(Fig. 1). 해당 MODIS 영상 자료 파일은 약 5.6 MB의 크기로, 각 격자의 위경도와 대기 자료 및 자료의 품질 자료 등 2차원과 3차원 자료를 포함한 58개의 변수가 저장되어 있다. 이중 10개의 변수에 대하여 GDAL의 처리 속도를 비교하였다(Table 1).

2.2. 분석도구

HDF 자료를 처리하기 위하여 GDAL 라이브러리를 사용하는 readGDAL_HDF를 구현하였다(Fig. 2).

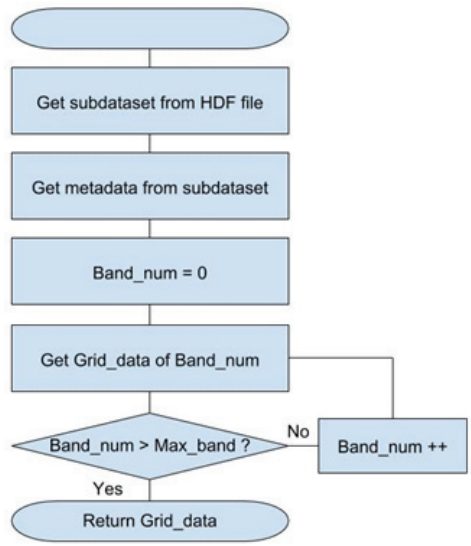

Fig. 2. The flow chart of readGDAL_HDF function.

해당 프로그램은 처리하고자 하는 파일명과 변수명을 입력값으로 사용한다. readGDAL_HDF의 주요 기능은 HDF파일에 저장되어 있는 개별 변수의 격자 자료를 읽어 컴퓨터의 메모리에 저장하는 것이다. readGDAL_HDF를 구현하기 위해 C++ 언어를 사용하였다.

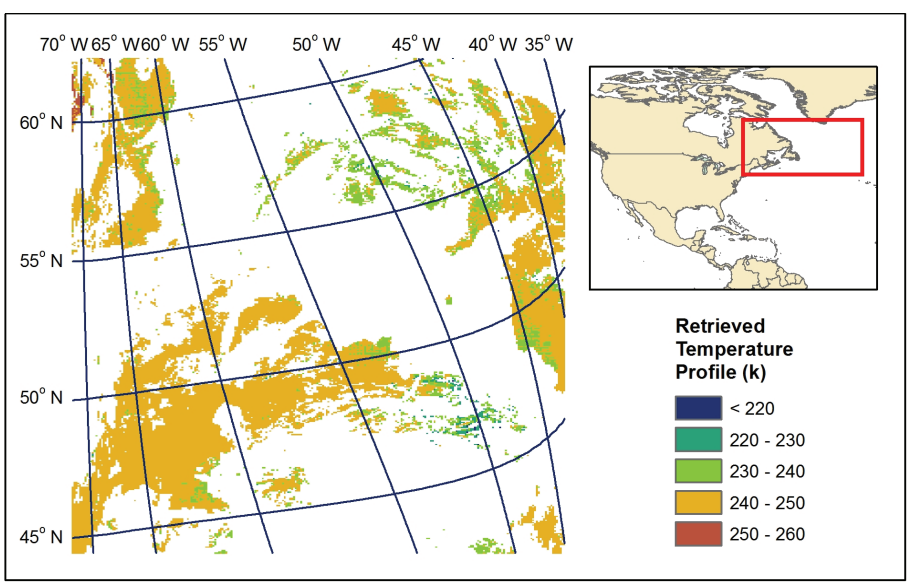

Fig. 1. The first layer of Retrieved Temperature Profile of the MYD07L2 data. This MODIS image contains the observation at 05:55 on July 19 in 2017.

GDAL 라이브러리의 설치 조건 별 readGDAL_HDF의 구동시간을 측정하였다. 운영체제, 배포 방식, 컴파일러 및 컴파일 옵션에 따라 GDAL 라이브러리의 설치 조건을 설정하였다(Table 2). 리눅스 시스템인 openSUSE 42.2 버전과 Ubuntu 12.04 버전을 사용하여 운영체제에 따른 속도 차이를 비교하였다. Ubuntu는 상대적으로 많이 사용되는 리눅스 시스템으로, 다양한 패키지들과 활용 예시들이 제공되는 장점이 있다. 한편 openSUSE는 안정성이 높으며, 그래픽 인터페이스를 통해 시스템 관리가 가능하다.

특정 컴파일러 사용에 따른 구동시간을 비교하기 위해 Intel 컴파일러 및 GNU 컴파일러가 사용되었다. GNU 컴파일러는 무료로 제공되는 표준적인 컴파일러이며, Intel 컴파일러는 유상으로 제공되나 복잡한 계산의 경우 GNU 컴파일러 보다 좋은 성능을 보이는 것으로 알려져 있다(Almomany and Balachandran, 2014). 추가적으로, GDAL 라이브러리들을 구축할 때 MODIS 영상사료를 지원여부를 결정할 수 있는 --with-hdf4와 같은 설정 선택 사항에 따른 구동 속도를 비교하였다.

다양한 설치 조건에 따라 구축된 GDAL 라이브러리의 자료 추출 속도를 비교하기 위해 R 스크립트를 작성하였다(Fig. 3). R은 공개형 통계 프로그램으로, 패키지 형태로 추가적인 기능을 지원하며, 격자 자료 처리에도 활용되고 있다(Yoo *et al*., 2017). 평균적인 구동 속도를 측정하기 위해서는 반복 구동이 필수적이다. R은 스크립트 형태로 명령을 자동화할 수 있어 수동으로 입력할 때 발생할 수 있는 처리시간의 지연 문제를 방지할 수 있다. 본 연구에서는 microbenchmark 패키지의 microbenchmark 함수를 사용하여 readGDAL_HDF를 100회씩 반복 구동하였으며, 정해진 10개의 변수에 대하여 순차적으로 구동되었다. 각 변수의 격자 수에 따라 처리 속도에 차이가 있으므로, 격자 수에 따라 변수들을 분류하고, 자료의 종류별로 평균적인 처리 속도를 비교하였다.

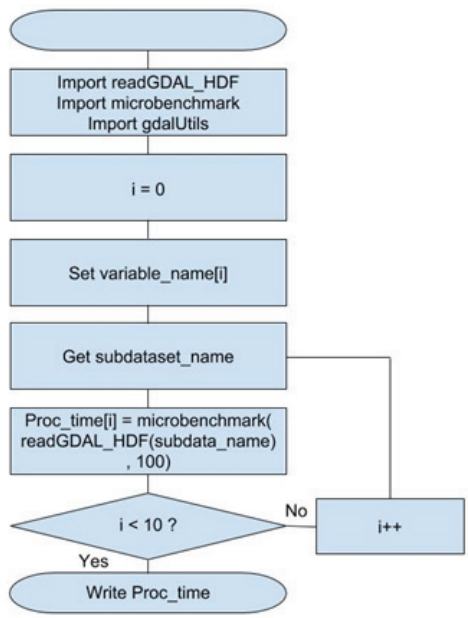

Fig. 3. The flow chart of the R script implemented to measure the execution time for the readGDAL HDF function.

Table 2. The environment and configuration under which the GDAL was built or installed to process a remote sensing data file in HDF

Identification code	Operating System[a]	Compiler[b]	Configuration Option[c]
suse_gcc	openSUSE	GNU	--with-hdf4
suse_icc	openSUSE	Intel	--with-hdf4
suse_nohdf4	openSUSE	GNU	--with-hdf4=no
suse_rpm*	openSUSE	-	-
ubu_gcc	Ubuntu	GNU	--with-hdf4
ubu_icc	Ubuntu	Intel	--with-hdf4
ubu_deb**	Ubuntu	-	-

[a] Operation system which GDAL installed.
[b] Compiler used to compile GDAL. GNU and Intel indicate GNU and Intel compiler respectively.
[c] Configuration options used when configuring GDAL.
* GDAL installed by RPM package manager.
** GDAL installed by Debian Package. GDAL version was 2.1.0 for the Debian package.

2.3. 전산자원

위성영상 자료 처리 속도 비교를 위해 대용량 자료 처리가 가능한 워크스테이션을 사용하였다. 워크스테이션은 Intel(R) Core(TM) i7-5820K 프로세서와 32GB 메모리가 탑재되었다. 여러 운영체제에서 처리 속도의 차이를 비교하기 위해, 동일한 하드웨어에서 각각의 운영체제를 설치하는 대신, 분석대상 운영체제를 가상머신에 설치하여 처리시간 분석을 수행하였다. 가상머신은 하드웨어 없이 컴퓨팅 환경을 제공하는 소프트웨어로, 이를 통해 단일 컴퓨터에서 여러 운영체제를 구동할 수 있다. 가상머신 관리용 어플리케이션으로 VirtualBOX (Oracle, Redwood City, CA, USA)를 사용하였다. VirtualBOX는 공개 소프트웨어로 Microsoft Windows, Linux, MacOS 등 다양한 운영체제에서 가상머신을 사용할 수 있게 한다.

Ubuntu 와 OpenSUSE가 동일한 계산 환경을 가지면서, 파일 처리에 충분한 정도의 자원을 가질 수 있도록 각각의 가상머신에 2개의 프로세서 코어 및 2GB의 메모리를 사용하도록 설정하였다. 바이너리 파일 배포판이 사용되는 가상머신의 경우, OpenSUSE의 공식 소프트웨어 센터(https://software.opensuse.org/package/gdal)와 Ubuntu GIS Personal Package Archives (http://ppa.launchpad.net/ubuntugis/ppa/ubuntu)에서 GDAL의 패키지 버전을 다운로드 받아 설치하였다. 소스코드를 컴파일하여 GDAL 라이브러리를 설치한 가상머신에는 GDAL 공식 홈페이지(http://www.gdal.org) 로부터 패키지로 설치 가능한 GDAL과 같은 버전의 소스코드(2.0.1 버전)를 다운로드 받아 이용하였다.

III. 결과 및 고찰

운영체제에 따라 MODIS 영상자료를 처리하는 시간은 자료의 차원에 따라 다른 양상을 보였다(Fig. 4 and Fig. 5). 2차원 MODIS 영상자료의 경우, openSUSE와 Ubuntu가 유사한 처리 속도를 보이는 것으로 나타났다. 예를 들어, 각 운영체제에서 빠른 속도를 보인 ubu_icc와 suse_gcc는 각각 15.28 및 15.73 milliseconds (ms)의 시간이 걸렸다. 자료 처리량이 많았던 3차원 MODIS 영상자료의 경우에는 Ubuntu에서 상대적으로 높은 빠른 속도로 처리하였다. 특히, 3차원 자료 처리를 위해 ubu_icc는 228ms가

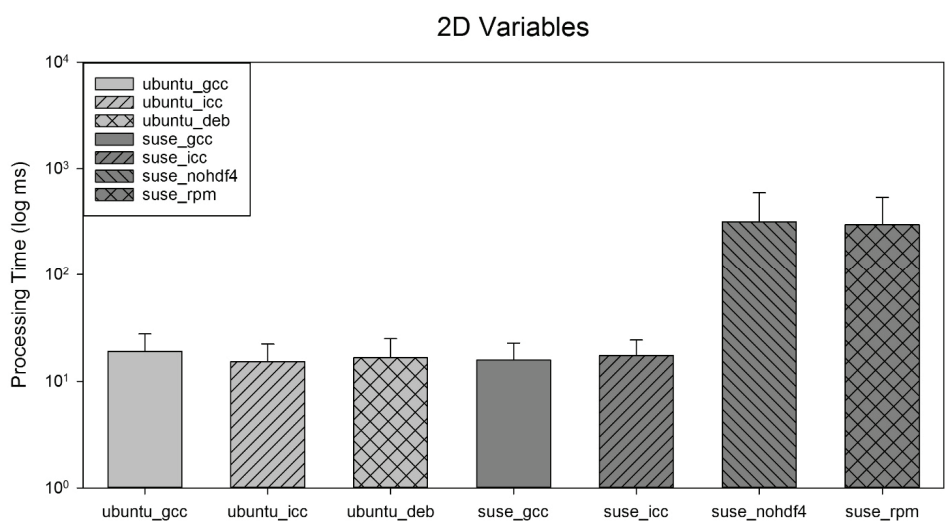

Fig. 4. The wall clock time to read 2-D variables contained within a MOD07 product data file in HDF using the readGDAL_HDF tool. The tool was built linking agianst the GDAL compiled under different operating systems, distribution package, compilers, and configuration. ubuntu and suse indicate Ubuntu and OpenSUSE operating systems, respectively. gcc and icc represent GNU and Inter compilers, respectively. Deb and rpm represent the GDAL package distributed by Ubuntu and OpenSUSE, respectively. nohdf4 denotes the --with-hdf4=no option for the compiler.

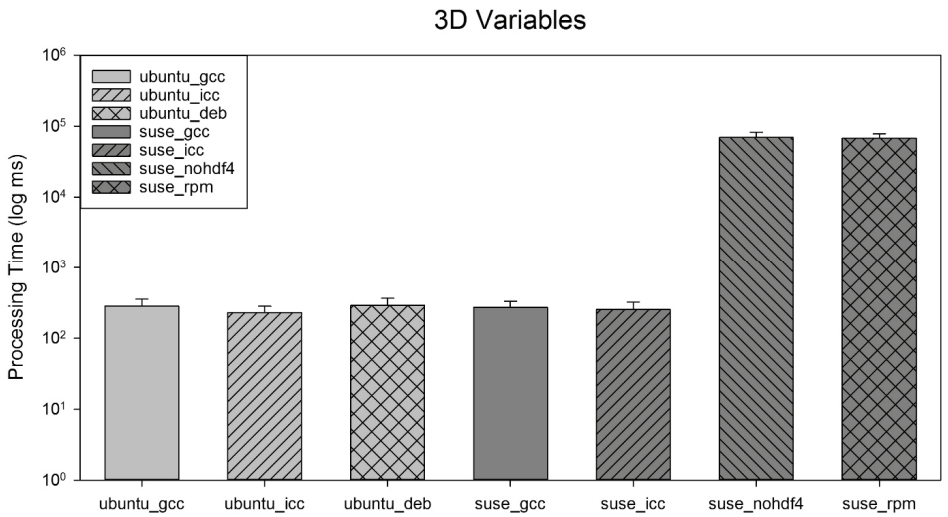

Fig. 5. The wall clock time to read 3-D variables contained within a MOD07 product data file in HDF using the readGDAL_HDF tool. The tool was built linking agianst the GDAL compiled under different operating systems, distribution package, compilers, and configuration. ubuntu and suse indicate Ubuntu and OpenSUSE operating systems, respectively. gcc and icc represent GNU and Inter compilers, respectively. Deb and rpm represent the GDAL package distributed by Ubuntu and OpenSUSE, respectively. nohdf4 denotes the --with-hdf4=no option for the compiler.

소요된 반면, suse_icc는 253ms가 걸려 10% 정도의 처리 시간 차이가 나타났다.

GDAL의 소스코드를 컴파일하여 구축된 라이브러리를 사용하였을 때, 운영체제와 컴파일러에 따라 처리 속도에 차이를 보였다. Ubuntu에서는 Intel 컴파일러를 사용하여 GDAL 라이브러리를 구축하였을 경우에 GNU 컴파일러를 사용하였을 때에 비해 2차원 및 3차원 영상자료를 보다 빠르게 처리하였다. 반면, openSUSE에서는 단순한 자료의 처리에는 GNU 컴파일러를 사용하였을 때 Intel 컴파일러에 비해 처리 속도를 높일 수 있었다. 3차원 영상자료 처리에는 OpenSUSE에서도 Intel 컴파일러가 더 신속하게 자료처리를 수행하였다.

GDAL의 설치 옵션이 HDF 처리 속도에 영향을 미치는 것으로 나타났다. GDAL을 --with-hdf4=no 옵션으로 컴파일한 경우 --with-hdf4 옵션에 비해 2차원 자료와 3차원 자료는 각각 19배 및 246배 느린 속도를 보였다. 반면, Ubuntu에서는 해당 옵션으로 컴파일한 경우, HDF 파일을 처리하지 못하는 것으로 나타났다. 이는 openSUSE의 경우 GDAL 라이브러리에 연결된 netCDF라이브러리가 HDF4를 지원하도록 컴파일 되어, 이를 통해 처리할 수 있었다. 반면, Ubuntu에서는 이러한 기능이 지원되지 않기 때문으로 추정되었다.

운영체제에 지원되는 패키지를 사용하였을 때, Ubuntu에서 openSUSE보다 월등히 빠른 처리 속도를 보였다. 패키지 버전의 경우 ubuntu_deb는 소스 버전과 비슷한 속도를 보였다. 반면, suse_rpm은 다른 처리에 비해 2차원 및 3차원 자료에서 모두 느린 것으로 나타났다. 특히, suse_rpm에서 얻어진 처리 속도는 --with-hdf4=no 옵션을 사용하였을 때와 유사하게 나타났다. 따라서 해당 패키지가 --with-hdf4=no 설정으로 구축되었을 것으로 추정되었다. 그러나, 패키지가 구축될 때 사용된 GDAL 라이브러리 구축에 사용된 옵션에 대한 정보가 제공되지 않아 이를 확인하지는 못하였다.

본 연구 결과에 따르면 MODIS 영상자료의 처리속도 면에서 Ubuntu를 사용하여 분석 시스템 구축하는 것이 유리한 것으로 나타났다. Ubuntu의 경우, 2차원 및 3차원 영상 자료 처리시 Intel 컴파일러로 구축하거나 deb 패키지로 배포된 GDAL 라이브러리를 사용하였을 때 상대적으로 빠른 시간 안에 자료처리를 완료하였다. 만약, openSUSE를 활용하고자 한다면, GNU

컴파일러를 선택하는 것이 동일한 하드웨어에서 더 높은 성능을 얻을 수 있었다.

운영체제에 상관없이 MODIS 영상자료의 처리 속도를 높이고자 한다면, 소스코드를 컴파일하여 GDAL 라이브러리를 사용하는 것이 유리할 것으로 보인다. Linux 운영체제와 함께 제공되는 패키지들을 사용할 경우, openSUSE의 경우와 같이 GDAL 라이브러리가 구축된 조건을 파악하기 어려워 상대적으로 처리속도가 느린 시스템이 구축될 수 있다. Ubuntu의 경우에는, 패키지 버전과 소스 컴파일 버전의 GDAL의 속도 차이가 openSUSE에 비해 적었다. 그러나, 패키지 버전이 컴파일 된 GDAL보다는 느린 처리 속도를 보여, 두 운영체제 모두에서 컴파일을 통한 GDAL 라이브러리 구축이 유리함을 보였다(Fig. 3 and Fig. 4). 패키지 버전의 경우 HDF 파일에 특화되도록 컴파일 되어 있지 않을 수 있다. 예를 들어, OpenSUSE에서는 rpm 패키지를 사용하여 처리하였을 때, netCDF 형식의 격자자료에 대한 호환성을 높이는 방향으로 GDAL 라이브러리가 구축되어 있었다. 패키지 버전의 경우, 배포자의 의도에 따라 라이브러리가 컴파일 되므로 사용자가 원하는 옵션이 반영되지 않을 가능성이 있다. 따라서, GDAL 라이브러리를 deb이나 rpm 과 같은 배포판을 사용하여 설치할 경우, 컴파일 옵션을 확인하여 특정 형식의 파일을 신속하게 처리할 수 있는 지에 대한 여부를 파악하는 것이 중요할 것이다.

인공위성 영상자료는 다양한 형식의 격자자료 형태로 제공된다. 예를 들어, 고해상도의 위성 자료인 Sentinel 자료의 경우 SAFE 형식의 파일로 제공된다. 30m 수준의 고해상도를 가진 영상자료를 제공하는 Landsat 자료의 경우 GEOTIFF 와 HDF 형식으로 제공되고 있다. 또한, 지표 반사 자료 등을 제공하는 MODIS 자료도 HDF와 GEOTIFF 형식으로 제공된다. 이들 자료들이 가진 형식에 따라, GDAL 라이브러리와 같은 영상자료 처리 라이브러리의 속도에 차이가 발생할 수 있다. 따라서, 인공위성 영상자료를 분석하기 위한 응용 프로그램을 개발하거나 설치할 때, 주로 사용되는 자료 형식을 빠르게 처리할 수 있는 라이브러리에 대한 성능을 파악하고 이에 대한 연구 결과를 공유하여 주어진 하드웨어 전산자원으로 보다 빠른 처리속도를 얻을 수 있을 것이다.

추후 연구에서는 인공위성 자료 처리 전 과정에 대한 성능 평가가 필요하다. MODIS swath 자료를 활용하기 위해, 자료 추출과 더불어 좌표계 변환, 파일간 중복 지역에 처리, 격자자료 모자이크 등의 과정이 진행된다(Li *et al*., 2010). 이러한 과정은 상당한 시간이 소요되므로, 최적화를 통해 성능 향상이 요구된다. 특히, 여러 프로세서를 활용하여 동시에 자료를 처리하는 병렬화를 통해 속도를 향상시킬 수 있을 것으로 생각된다. 예를 들어, openMP (open multi-Processing) 나 MPI (Message Passing Interface) 등의 병렬화 지원 라이브러리를 통해 격자형 자료 처리를 병렬화 하면 속도를 향상시킬 수 있다. 예를 들어 Yoo *et al*.(2017)은 격자형 기상 자료 처리를 위해 openMP를 활용한 병렬화를 통해 5 배 빠른 처리 속도를 얻었다. Tie *et al*.(2018)은 MPI를 사용하여 MODIS 영상자료 중 지표 온도 자료의 처리 속도를 높였다.

자료 처리를 위해 병렬계산을 수행하는 경우, 처리 속도에 대한 운영체제 및 컴파일러의 영향이 단일 프로세서만 사용한 본 연구와는 다르게 나타날 수 있다. 특히, 병렬처리 라이브러리의 경우 컴파일러의 최적화 옵션에 따라 처리 속도에 영향이 있는 것으로 알려져 있다(Tristram and Bradshaw, 2012). 또한 본 연구에서는 단일 프로세서를 사용하여 2GB의 시스템 메모리가 설정된 가상머신으로 처리가 가능하였으나, 동시에 다수의 자료를 처리하는 병렬처리의 경우 대용량의 메모리가 요구될 수 있다. 따라서, 추후 병렬화에 따른 영상자료 처리 최적화 지원을 위해 컴파일러 옵션과 사용하는 프로세서의 수에 따른 처리 속도 및 메모리 요구량에 대한 분석이 필요하다.

본 연구의 결과는 다수의 영상 자료 처리가 요구되는 지역 단위의 작물 생육 추정 시스템의 성능을 향상시킬 수 있도록 지원할 것이다. 운영체제와 컴파일러 선택에 따라 2~24%의 속도 차이가 있는 것으로 나타나, 동일한 하드웨어 조건이라도 보다 효율적인 시스템을 구성할 수 있을 것으로 판단된다. 예를 들어, 1개의 3차원 변수에 대해 미국 전역의 1년치 자료인 365일 x 23개의 swath 자료를 처리한다면, 조건에 따라 데이터 추출과정에서만 약 6분 정도의 차이를 보일 수 있다. 또한, 패키지 형태의 라이브러리로 시스템을 구성하는 것은 소스를 컴파일 하는 것에 비해 자료 처리 측면에서 낮은 성능을 보일 수 있어 주의해야 한다. 최근, 고해상도의 인공위성 자료나, 무인기로부터 얻어진 초고해상도의 영상을 바탕으로 작물의 생육을 감시 및 추정하는 연구들이 진행되고 있다(Lee *et al*., 2016). 무인기 영상의 경우, 인공위성 자료에 비해 저비용으로 농가 단위의 자료를 생산할 수 있어 앞으

로 무인기 영상이 농업 생산성 파악에 많이 활용될 것으로 추정된다. 따라서, 인공위성 자료 뿐 아니라 무인기로부터 얻어진 초고해상도 영상 자료를 처리하기 위한 도구들을 최적화할 수 있는 연구가 필요할 것으로 보인다.

적 요

지역이나 전구 규모의 농업 생태계를 감시하기 위해 HDF 형식으로 제공되는 MODIS 원격 탐사자료가 사용되어 왔다. 대개의 경우, 다량의 영상자료들이 처리되어야 하기 때문에, 이들 자료의 처리 성능을 향상시키는 것이 유리하다. 본 연구는 HDF 파일을 처리할 수 있는 GDAL과 같은 라이브러리가 운영 체제나 배포 방식 등에 따른 처리속도의 차이를 확인하여 원격탐사 자료 처리 시스템 구축을 지원하고자 하였다. 이를 위해, GDAL이 시스템에 설치되는 주요 조건들에 따라 MODIS 영상자료 처리 시간을 측정하고 비교하였다. 운영 체제(Ubuntu 및 openSUSE), 컴파일러(GNU 및 Intel), 설치 옵션 및 바이너리 패키지 조건을 조합하여 GDAL성능 비교가 이루어졌다. 각 조건에 따라 설치된 GDAL을 사용하여 MODIS 영상 중 대기측정 자료(MOD07)의 2차원 변수와 3차원 변수에 해당하는 총 10 종의 자료를 추출하였다. 자료처리에 소요된 구동 시간은 각 변수 값을 시스템 메모리에 저장하는 작업이 끝난 직후 측정되었다. 가장 좋은 성능을 보인 설치 조건은 Ubuntu에서 Intel Compiler를 사용하여 컴파일 된 GDAL을 사용하는 것이었다. OpenSUSE에서는 GNU와 Intel 컴파일러가 각각 2차원 자료와 3차원 자료를 처리하기 위한 작업에 효과적인 것으로 나타났다. 한편 “--with-hdf4=no” 옵션으로 컴파일 된 GDAL과 RPM package manager 버전의 GDAL의 경우, 다른 조건에 비해 상당히 낮은 성능을 보였다. 이러한 결과는 운영 체제나 컴파일러, 설치 옵션 등을 조정하여 원격 탐사자료 처리 도구의 속도를 개선할 수 있다는 것을 암시하였다. 특히, 원격 탐사 자료의 경우 다양한 형식으로 배포되므로, 이를 처리하는 라이브러리들이 최고의 성능을 발휘할 수 있는 조건을 탐색하고 이러한 결과의 공유가 후속연구에서 진행되어야 할 것으로 보인다.

감사의 글

본 논문은 농촌진흥청 공동연구사업 (과제번호: PJ013452042018)의 지원에 의해 이루어진 것임.

REFERENCES

Almomany, A., A. Alquraan, and L. Balachandran, 2014: GCC vs. ICC comparison using PARSEC Benchmarks. *International Journal of Innovative Technology and Exploring Engineering* **4**(7).

Andrew, M. E., M. A. Wulder, and T. A. Nelson, 2014: Potential contributions of remote sensing to ecosystem service assessments. *Progress in Physical Geography* **38**(3), 328-353.

Ban, H.-Y., K. S. Kim, N.-W. Park, and B.-W. Lee, 2016: Using MODIS data to predict regional corn yields. *Remote Sensing* **9**(1), 16pp.

Cohen, W. B., and S. N. Goward, 2004: Landsat’s role in ecological applications of remote sensing. *Bioscience* **54**(6), 535-545.

Busetto, L., and L. Ranghetti, 2016: MODIStsp : An R package for automatic preprocessing of MODIS Land Products time series. *Computers & Geosciences* **97**, 40-48.

Doraiswamy, P., J. Hatfield, T. Jackson, B. Akhmedov, J. Prueger, and A. Stern, 2004: Crop condition and yield simulations using Landsat and MODIS. *Remote sensing of environment* **92**(4), 548-559.

Hong, S. Y., S.-I. Na, K.-D. Lee, Y.-S. Kim, and S.-C. Baek, 2015: A study on estimating rice yield in DPRK using MODIS NDVI and rainfall data. *Korean Journal of Remote Sensing* **31**(5), 441-448.

Jobbágy, E. G., O. E. Sala, and J. M. Paruelo, 2002: Patterns and controls of primary production in the Patagonian steppe: a remote sensing approach. *Ecology* **83**(2), 307-319.

Lee, K.-D., S.-I. Na, S.-Y. Hong, C.-W. Park, K.-H. So, and J.-M. Park, 2017: Estimating corn and soybean yield using MODIS NDVI and meteorological data in Illinois and Iowa, USA. *Korean Journal of Remote Sensing* **33**(5), 741-750.

Lee, J.-H., S.-K. Kang, K.-C. Jang, J.-H. Ko, and S.-Y. Hong, 2011: The evaluation of meteorological inputs retrieved from MODIS for estimation of gross primary productivity in the US corn belt region. *Korean Journal of Remote Sensing* **27**(4), 481-494.

Li, J., M. Humphrey, C. Van Ingen, D. Agarwal, K. Jackson, and Y. Ryu, 2010: escience in the cloud: A modis satellite data reprojection and reduction pipeline in the windows azure platform. In *2010 IEEE International Symposium on Parallel & Distributed Processing (IPDPS)*, IEEE, 1-10.

Lobell, D. B., G. P. Asner, J. I. Ortiz-Monasterio, and T. L. Benning, 2003: Remote sensing of regional crop production in the Yaqui Valley, Mexico: estimates and uncertainties. *Agriculture, Ecosystems & Environment* **94**(2), 205-220.

Mourani, G.,2001: *Securing and Optimizing Linux: The Ultimate Solution*. Open Network Architecture, Inc., 855pp.

Prasad, A. K., L. Chai, R. P. Singh, and M. Kafatos, 2006: Crop yield estimation model for Iowa using remote sensing and surface parameters. *International Journal of Applied Earth Observation and Geoinformation* **8**(1), 26-33.

Tie, B., F. Huang, J. Tao, J. Lu, and D. Qiu, 2018: A parallel and optimization approach for Land-Surface Temperature retrieval on a Windows-Based PC cluster. *Sustainability* **10**(3), 621pp.

Tristram, W., and K. Bradshaw, 2012: Performance optimisation of sequential programs on multi-core processors. *Proceedings of the South African Institute for Computer Scientists and Information Technologists Conference*, Pretoria, South Africa, ACM, 119-128.

Turner, D. P., W. D. Ritts, W. B. Cohen, S. T. Gower, S. W. Running, M. Zhao, M. H. Costa, A. A. Kirschbaum, J. M. Ham, S. R. Saleska, and D. E. Ahl, 2006: Evaluation of MODIS NPP and GPP products across multiple biomes. *Remote Sensing of Environment* **102**(3-4), 282-292.

Vancutsem, C., P. Ceccato, T. Dinku, and S. J. Connor, 2010: Evaluation of MODIS land surface temperature data to estimate air temperature in different ecosystems over Africa. *Remote Sensing of Environment* **114**(2), 449-465.

Yoo, B. H., and K. S. Kim, 2017: Development of a gridded climate data tool for the COordinated Regional climate Downscaling EXperiment data. *Computers and Electronics in Agriculture* **133**, 128-140.