

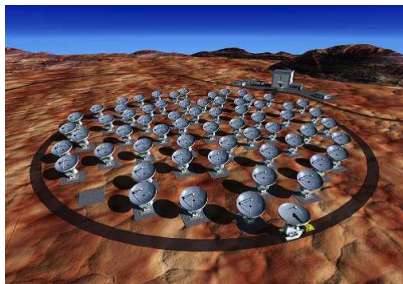
Interactive Web Viewer for ALMA Cube Data and Zero Time Endian Conversion Technique

Satoshi Eguchi

National Astronomical Observatory of Japan

2013 May 13

Introduction



- **Atacama Large Millimeter/submillimeter Array**
 - ~ 200 TB/year outputs as raw data
 - May exceed $\gtrsim 2$ TB as processed data cube for one target
-
- Internet bandwidth of the world average is 2.6 Mbps.
 - It will take ~ 75 days to download a full data cube!
 - To analyze ALMA data cube on one's *personal* computer is unrealistic.

Approach

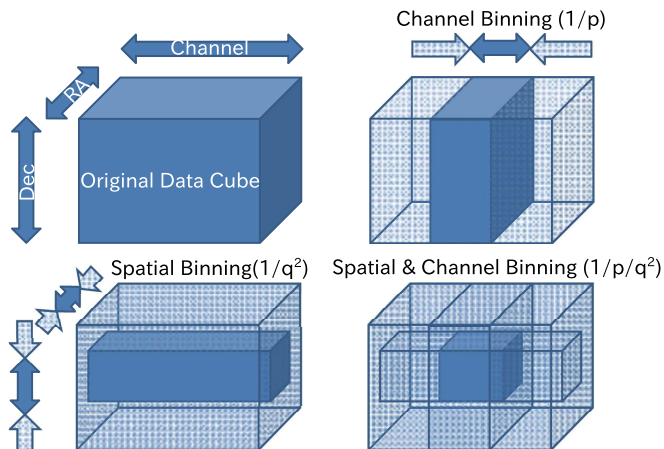
Fact

- What astronomers expect in an ALMA data cube is different from person to person.
- But all of them do not require the full data cube.

Our Approach

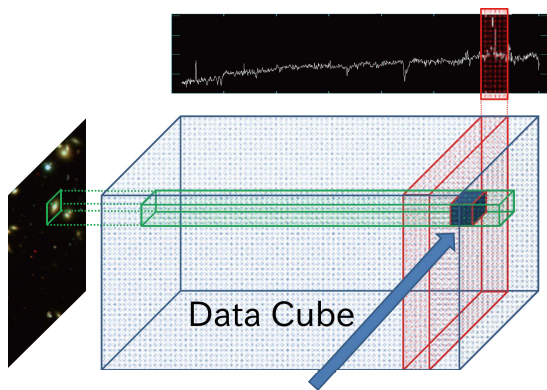
- Provide users with a web application which enables each user to extract and download what satisfies their own needs from a big data cube.
- Newly extracted data cube should be small, so
 - Save network traffic and our computation resources
 - Utilize users' local computation resources

Data Compression: Binning



The data cube size becomes pq^2 times smaller than original one.

Data Compression: Cut-Out



Cut out and download interested region only

Note

The data cube is already compressed by binning.

ALMAWebQL –Introduction–

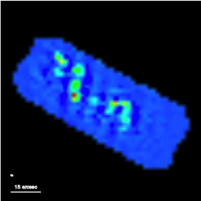
ALMAWebQL enables us to select desired **binning parameters** and a **cut-out region** **graphically** with a **web browser**.

QUICK LOOK SYSTEM FOR ALMA VLJ SERVICE

px0.vo.nao.ac.jp:8050/almawebql/ALMAWebQL.html?path=http://k31sk2f2f7focalhost531a8060k2f7skymodek2f-don2f7kapk2f/alma_int92fynock3FREQUESTK3DsdvQueryK26LANCK3DADQLK26QUERYK3DSELECTK2

File Information

Title	Data Set ID	Observation Date	File Size
Antennae	ALMA00000006	2011/06/28 12:07:32	96.44 kB



Resolution :

- ▼ Enumerate...
- 1.109 arcmin/pix
- 33.28 asec/pix
- 16.64 asec/pix
- 8.320 asec/pix
- 4.160 asec/pix

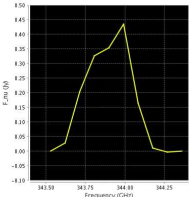
Zoom :

Log Scale

Zoom : x1

R.A. : 12h01m54.00s

Dec. : -18d53m09.81s



Resolution :

Res. : 92.276 MHz/ch

344.435 GHz

Force Reload File Download

ready.

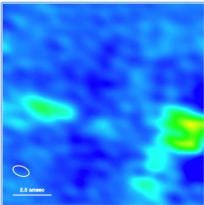
Change resolution, position and zoom...

QUICK LOOK SYSTEM FOR ALMA VU SERVICE

px0.v0.nao.ac.jp:8010 /almawebql.html?path=/kit3/627627fcca805d53a8060927skymode%2Fdo%2Fcap%2Falma_int%2Fsync%3FREQUEST%3DdoQuery%26LANCK%3DADQ%26QJERY%3DSELECT%26

File Information

- Title: Antennae
- Data Set ID: ALMA0000006
- Observation Date: 2011/06/28 12:07:32
- File Size: 150.3 MB

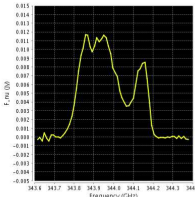


Resolution :
 Enumerate...
 2.080 asec/pix
 1.040 asec/pix
 520.0 mas/pix
 260.0 mas/pix
 130.0 mas/pix

Zoom :
 Enumerate...
 x1
 x2
 x4
 x8
 x16

Log Scale

RA. : 12h01m53.95s
 Dec. : -18d53m07.76s



Resolution :
 Enumerate...
 184.552 MHz/ch
 92.276 MHz/ch
 46.138 MHz/ch
 23.069 MHz/ch
 11.535 MHz/ch

Log Scale

343.921 GHz

[Force Reload](#) [File Download](#)

ready.

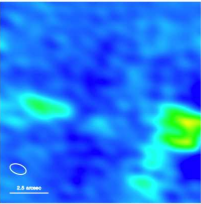
Select spectral range by mouse dragging...

QUICK LOOK SYSTEM FOR ALMA VUJ SERVICE

px0.vv.nao.ac.jp:8010 /almawebql.html?path=http://3.169.27.92/focalhost/3.180.60/27/skynode/27/don27/cap/27/alma_int27/feynck3/FREQQUEST/3DdoQuery/2/6/LANCK3DADQL/2/6/QUERY/3/DOSELECT/2

File Information

- Title: Antennae
- Data Set ID: ALMA0000006
- Observation Date: 2011/06/28 12:07:32
- File Size: 150.3 MB



Resolution:

Enumerate...

- 2.080 arcsec/pix
- 1.040 arcsec/pix
- 520.0 mas/pix
- 260.0 mas/pix
- 130.0 mas/pix

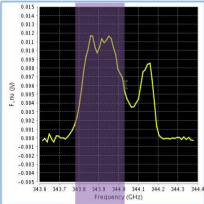
Zoom:

Enumerate...

- x1
- x2
- x4
- x8
- x16

Log Scale

RA: 12h01m53.95s
Dec: -18d53m07.76s



Resolution:

Enumerate...

- 184.552 MHz/ch
- 92.276 MHz/ch
- 46.138 MHz/ch
- 23.069 MHz/ch
- 11.535 MHz/ch

Log Scale

344.031 GHz

Force Reload File Download

ready.

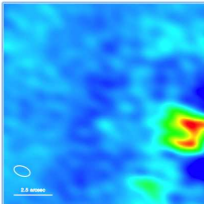
Select spectral range by mouse dragging...

QUICK LOOK SYSTEM FOR ALMA VUJ SERVICE

px0.v0.nao.ac.jp:8010 /almawebql/ALMAWebQL.html?path=http://3.146.27.92/focalhost53/ALMA0000006/2f1skymode/2f1d0n2f1cap/2f1alma_int/2f1sync/3FREQUENT/3DdoQuery/2/6LANCK/3DADQL/2/6QUERY/3DSELECT/2

File Information

- Title: Antennae
- Data Set ID: ALMA0000006
- Observation Date: 2011/06/28 12:07:32
- File Size: 150.3 MB

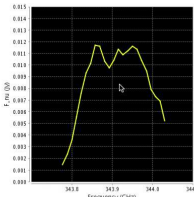


Resolution:
 Enumerate...
 2.080 asec/px
 1.040 asec/px
 520.0 mas/px
 260.0 mas/px
 130.0 mas/px

Zoom:
 Enumerate...
 x1
 x2
 x4
 x8
 x16

Log Scale

RA. : 12h01m53.95s
 Dec. : -18d53m07.76s



Resolution:
 Enumerate...
 184.552 MHz/ch
 92.276 MHz/ch
 46.138 MHz/ch
 23.069 MHz/ch
 11.535 MHz/ch

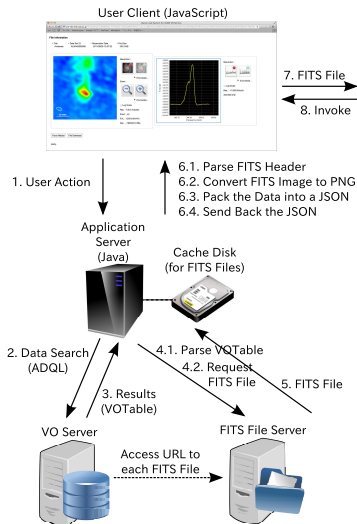
Log Scale

343.918 GHz

Force Reload File Download

ready.

Implementation



- Built on Google Web Toolkit
- Let a web browser do as many **but simple operations as possible** to reduce network traffic.
 - Coordinate transforms
 - Zoom
- Contact the VO server via TAP interface and retrieve the URL for a FITS file
- A FITS file is cached on the local disk of the application server.

File I/O Speed

- Hard Disk : ~ 50 MB/s
- Solid State Disk (SSD) : ~ 250 MB/s
 - A raid0 array consists of 16 SSDs reaches ~ 4 GB/s.
- Cached by OS (or RAM Disk) : ~ 10 GB/s
 - ← This is in case of ALMAWebQL.
- File reading time and computing time are comparable at ~ 10 GB/s.

We can expect performance gain by optimizing the treatment of FITS files.

Endian

▷ Little Endian

- $0x12345678 \rightarrow 78\ 56\ 34\ 12$
- ex) Intel, AMD

▷ Big Endian

- $0x12345678 \rightarrow 12\ 34\ 56\ 78$
- ex) SPARC, PowerPC

- **Java VM and FITS File : Big Endian**

Endian conversion is automatically performed inside FITS libraries!

Comparison of Endian Conversion Timing 1

We can implement the codes to sum up all elements in a FITS file in two different ways:

- Convert endian of all elements in advance, then compute the summation
→ on ahead endian conversion method
- Perform endian conversion and compute the summation at the same time in one loop
→ just-in-time endian conversion method

Comparison of Endian Conversion Timing 2

On Ahead Endian Conversion Method

```
{  
  for (size_t i = 0; i < len; ++i) {  
    v[i] = convert_endian(v[i]);  
  }  
  
  double sum = 0.0;  
  for (size_t i = 0; i < len; ++i) {  
    sum += v[i];  
  }  
}
```

Just-in-Time Endian Conversion Method

```
{  
  double sum = 0.0;  
  for (size_t i = 0; i < len; ++i) {  
    sum += convert_endian(v[i]);  
  }  
}
```

Benchmark 1

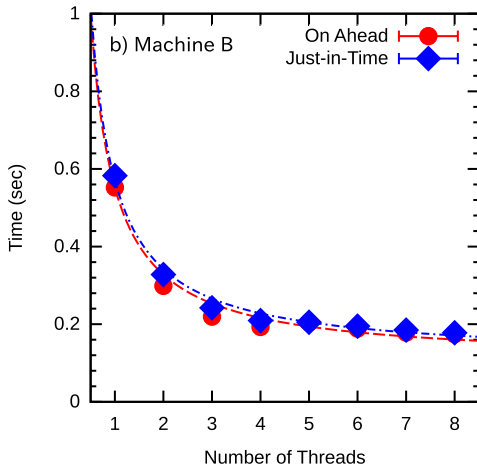
On Ahead Endian Conversion Method

```
{  
  for (size_t i = 0; i < len; ++i) {  
    v[i] = convert_endian(v[i]);  
  }  
  
  double sum = 0.0;  
  for (size_t i = 0; i < len; ++i) {  
    sum += v[i];  
  }  
}
```

Just-in-Time Endian Conversion Method

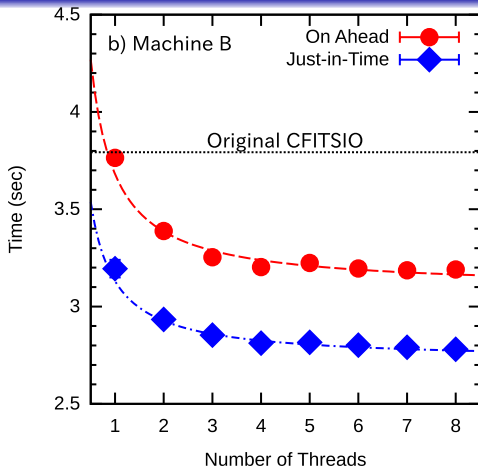
```
{  
  double sum = 0.0;  
  for (size_t i = 0; i < len; ++i) {  
    sum += convert_endian(v[i]);  
  }  
}
```

Benchmark 2



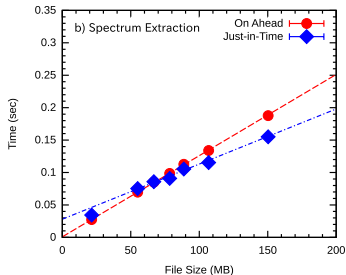
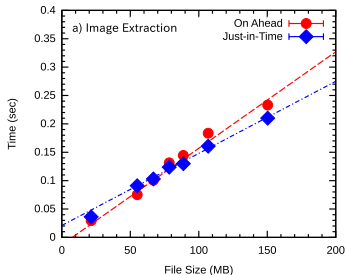
- We observed no significant difference of computation time between both methods (→ zero time endian conversion).

Benchmark 3



- But just-in-time method is **20% in single thread** and **40% in multi-thread faster** than on ahead method in total (including file reading time).

Application to ALMAWebQL



- We applied just-in-time endian conversion method to ALMAWebQL, and measured its file size dependency in single thread.
- Just-in-time method wins on ahead method above 100 MB.
- Just-in-time method is suitably 20% faster than on ahead method above 200 MB.

Summary

- ALMA data cubes may exceed 2 TB and it is unrealistic to download a full data cube.
- By providing a web application with intuitive graphical interface, an user can cut out their desired part of a data cube with desired resolution.
- From technical point of view of that application, file reading time, endian conversion time and data processing time are comparable.
- To gain more performance, we developed just-in-time endian conversion method.
- Just-in-time method is 20% in single thread and 40% in multi-thread faster than on ahead method, which is widely used inside FITS libraries, above 200 MB.