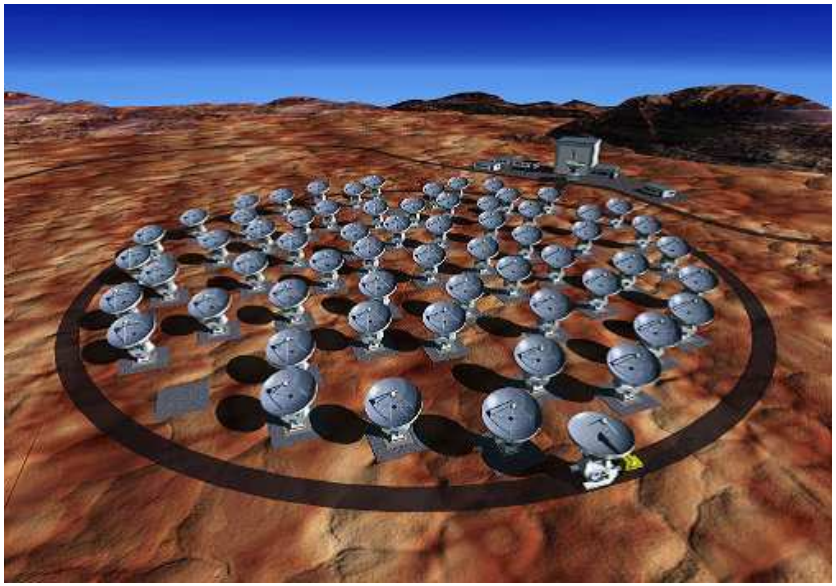
Interactive Web Viewer for ALMA Cube Data and Zero Time Endian Conversion Technique

Satoshi Eguchi

National Astronomical Observatory of Japan

2013 May 13

Introduction



- **A**tacama **L**arge **M**illimeter/submillimeter **A**rray
- $\sim$ 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 $\sim$ 75 days to download a full data cube!
- To analyze ALMA data cube on one's *personal* computer is unrealistic.

Approach

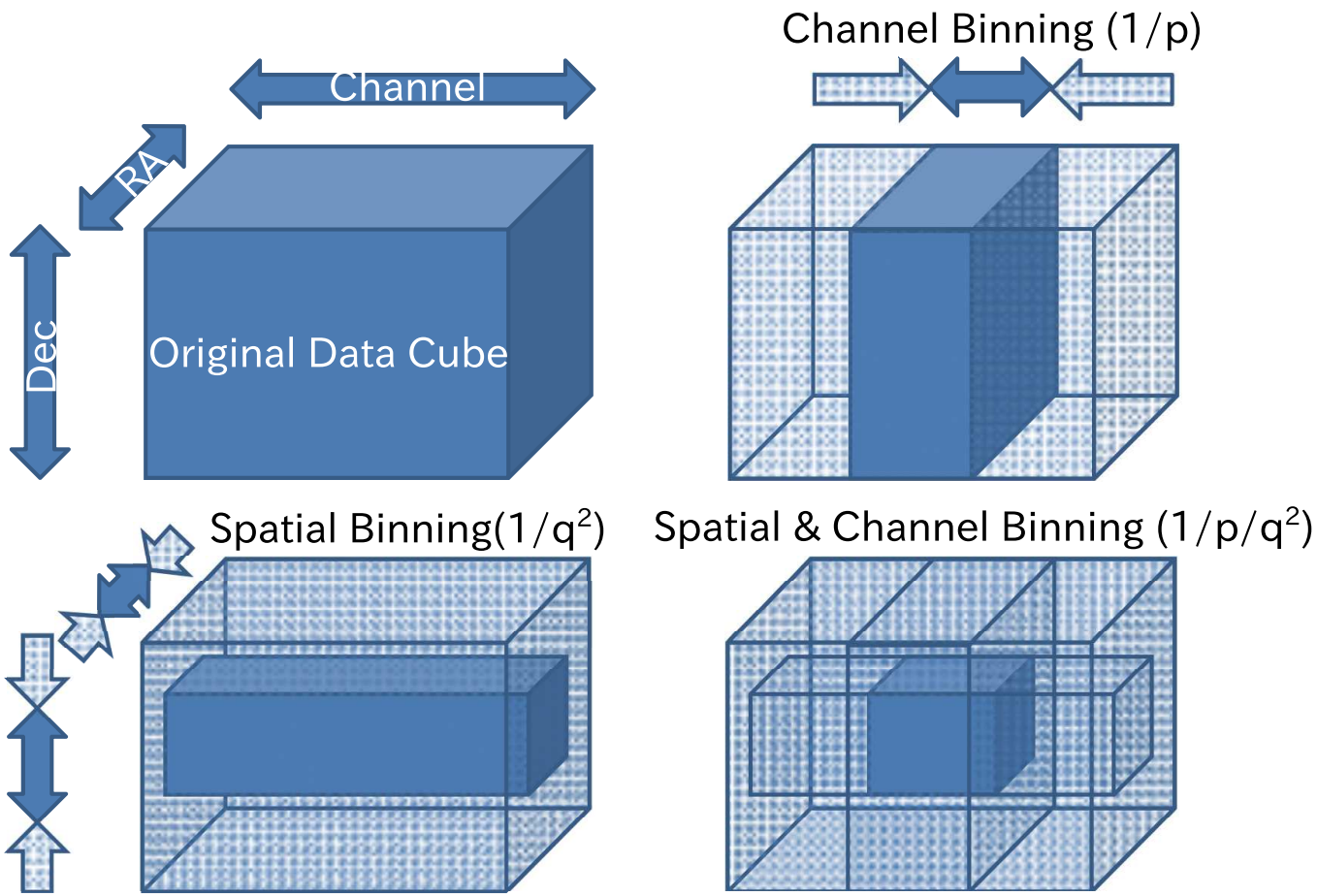
Fact

- 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

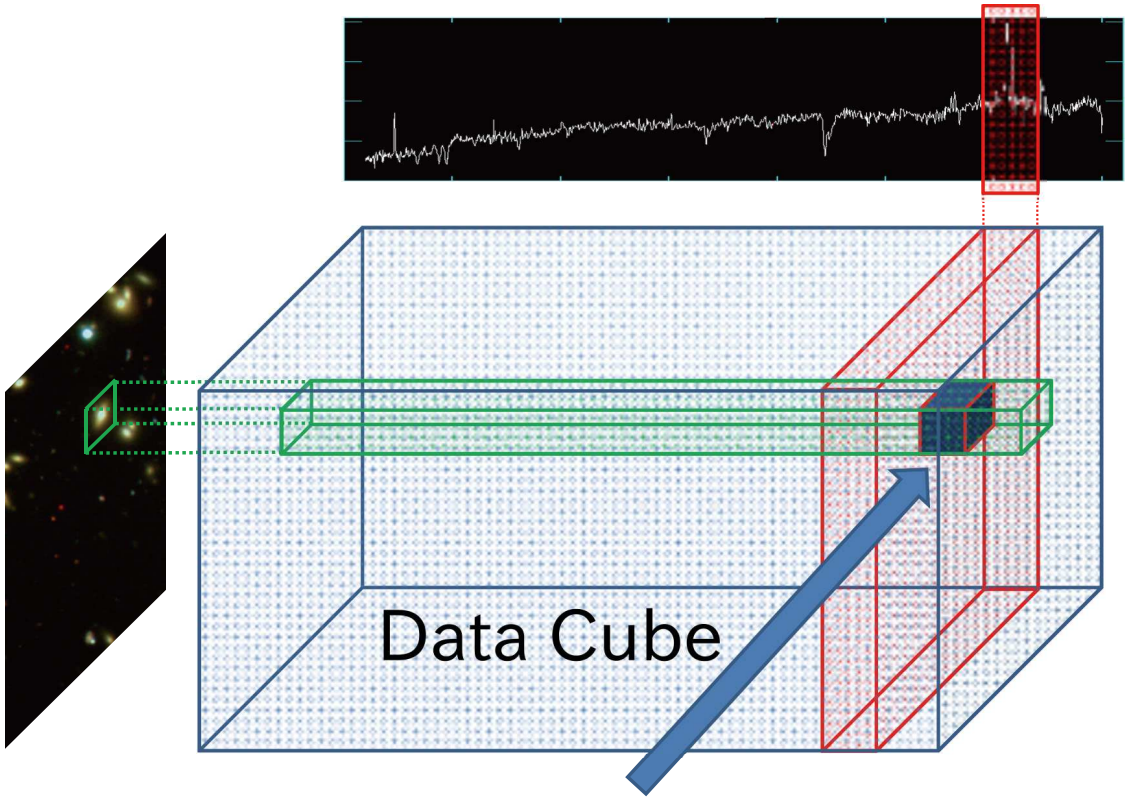
- Provide users with **a web application** which enables each user to extract and download what satisfies their own needs from a big data cube.
- Newly extracted data cube should be small, so
  - Save network traffic and our computation resources
  - Utilize users' local computation resources

Data Compression: Binning



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

Data Compression: Cut-Out



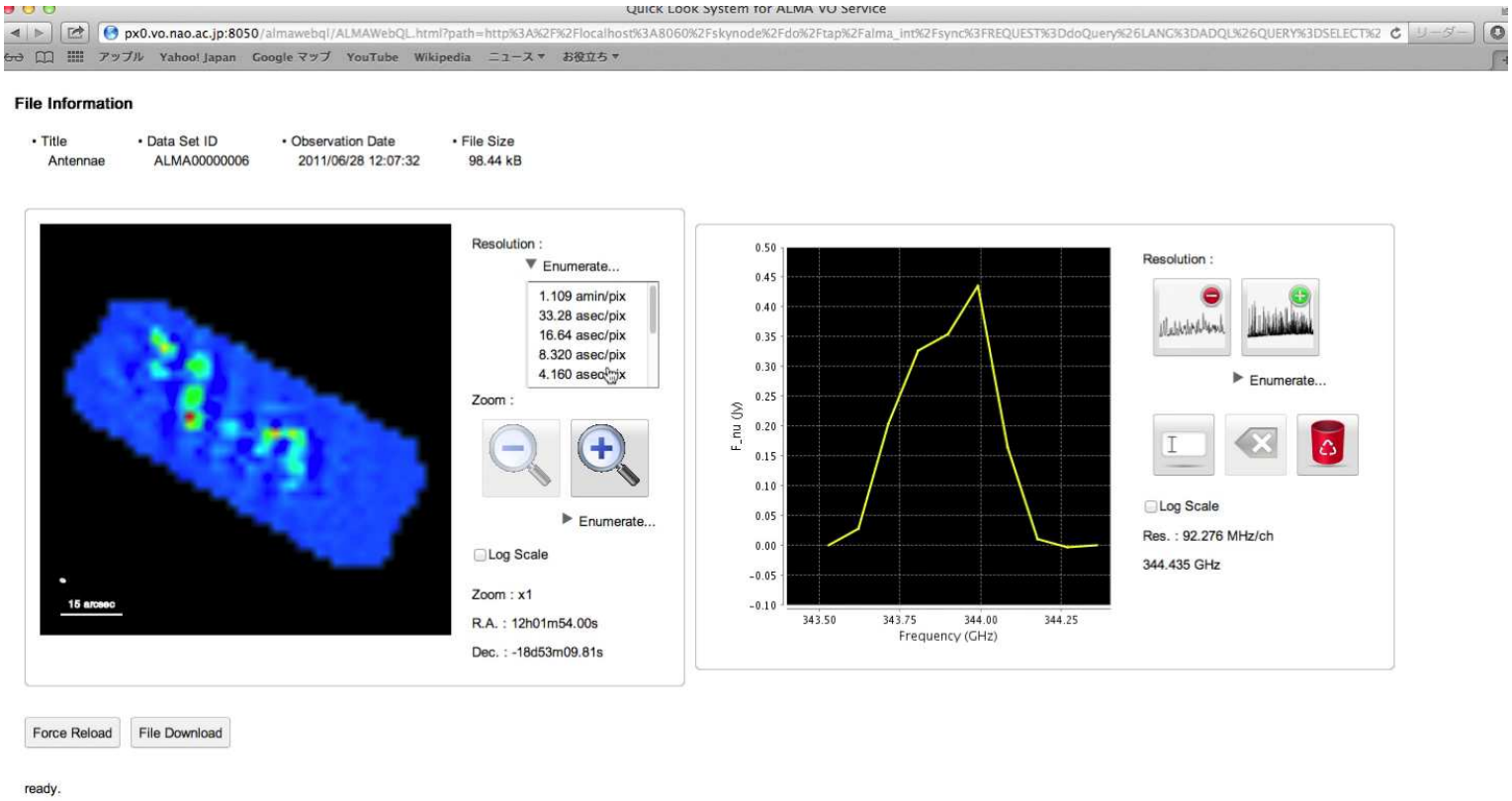
Cut out and download interested region only

Note

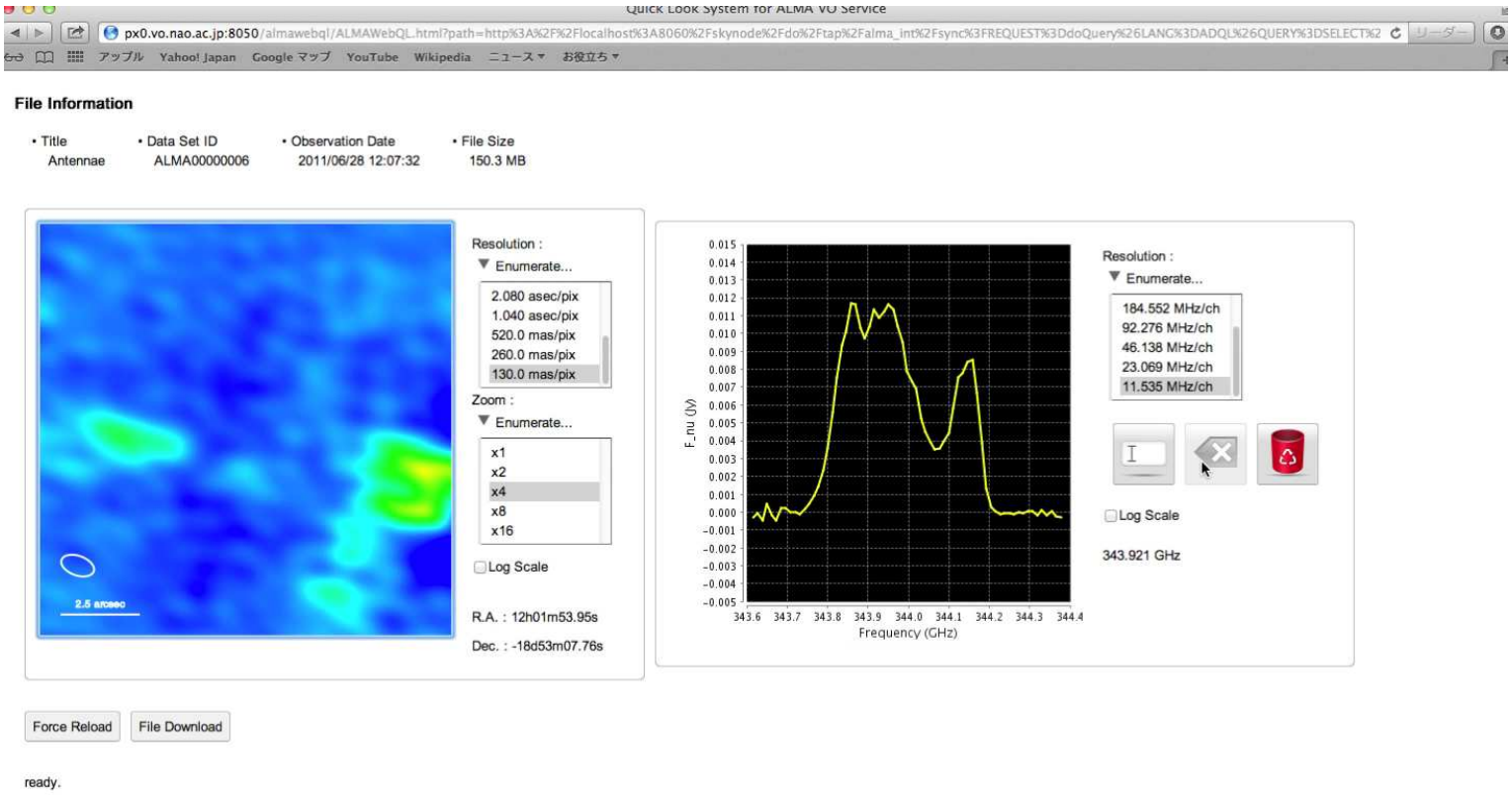
The data cube is already compressed by binning.

ALMAWebQL –Introduction–

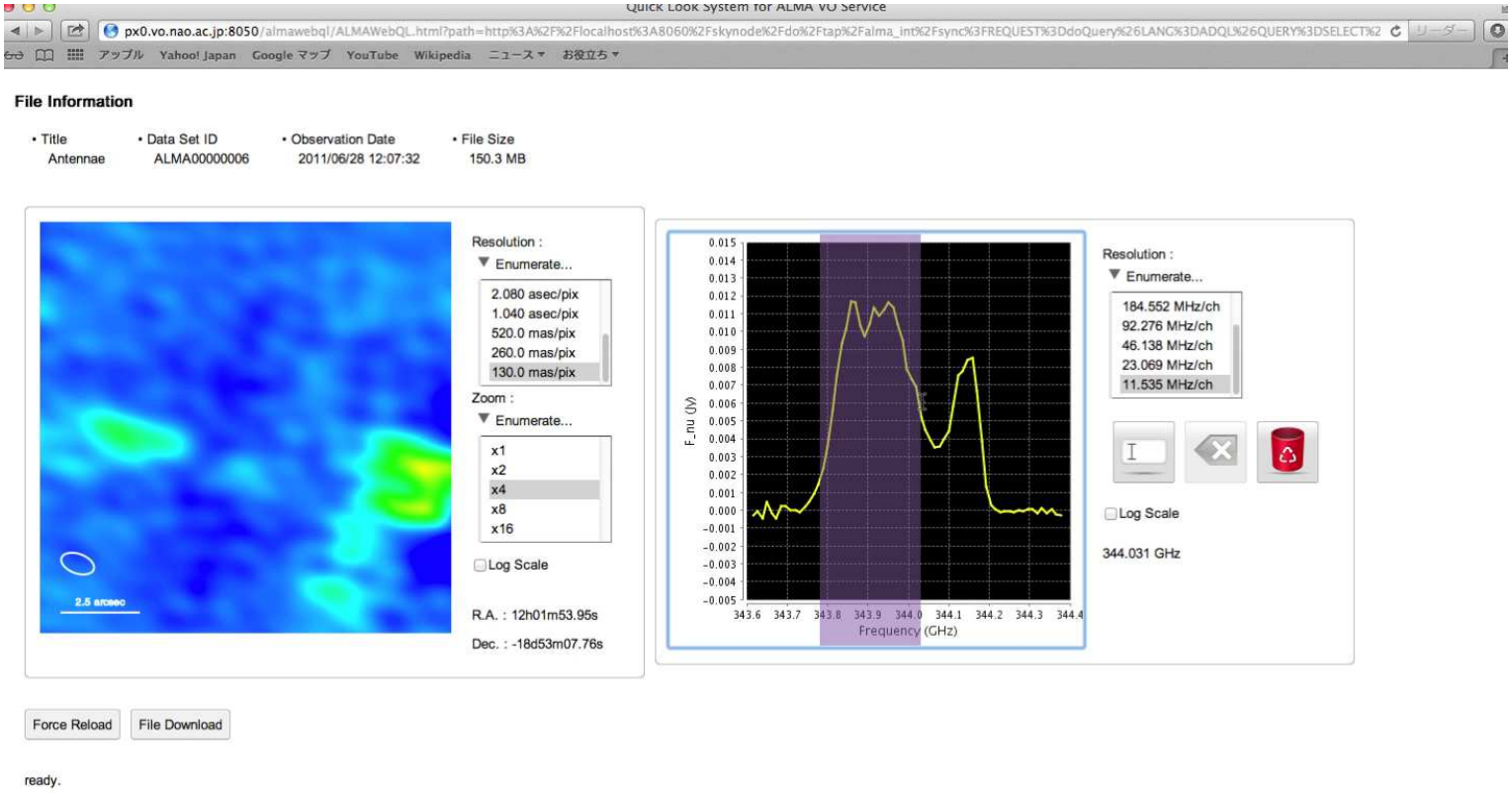
ALMAWebQL enables us to select desired **binning parameters** and **a cut-out region** graphically with a web browser.



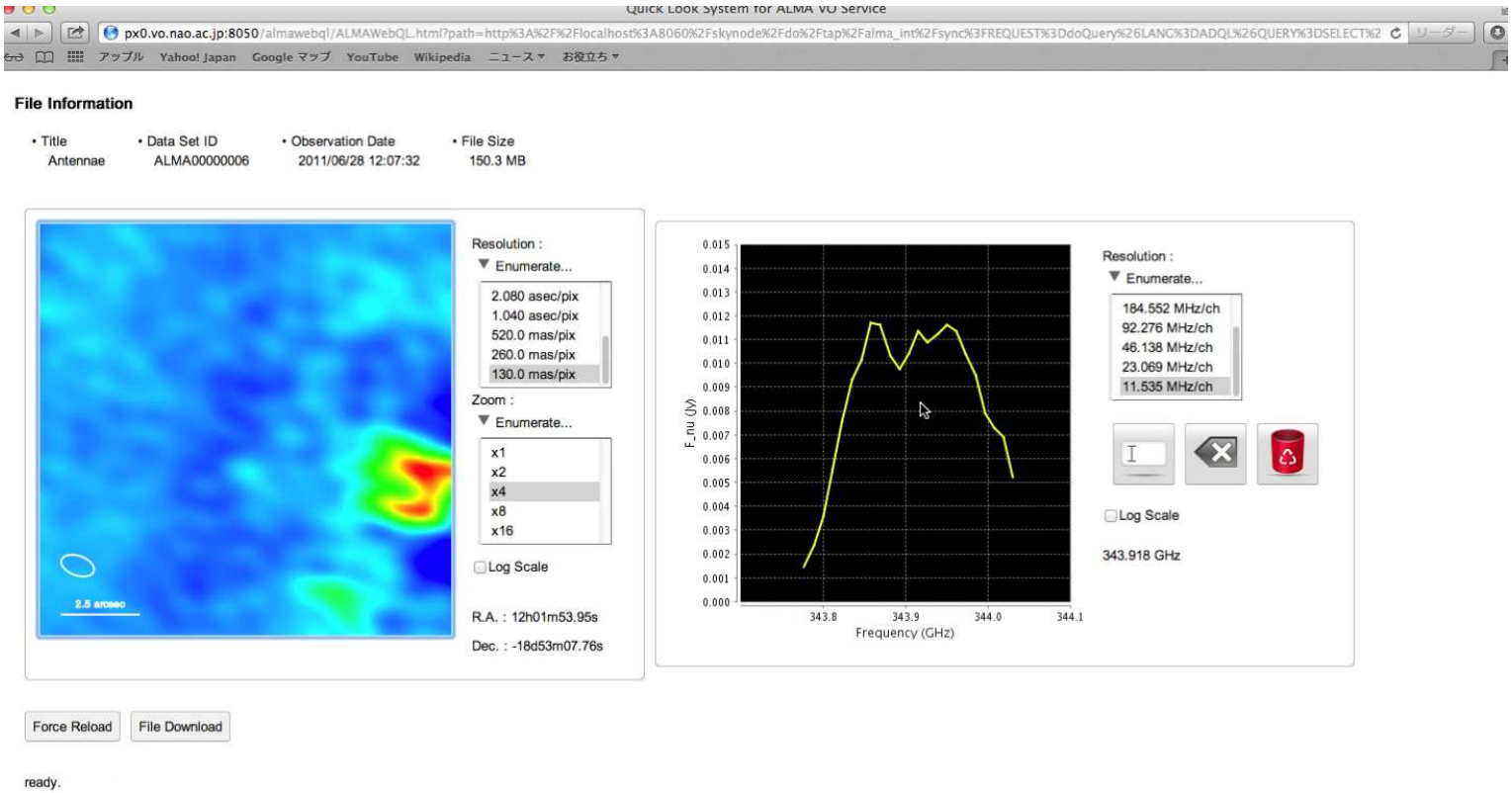
Change resolution, position and zoom...



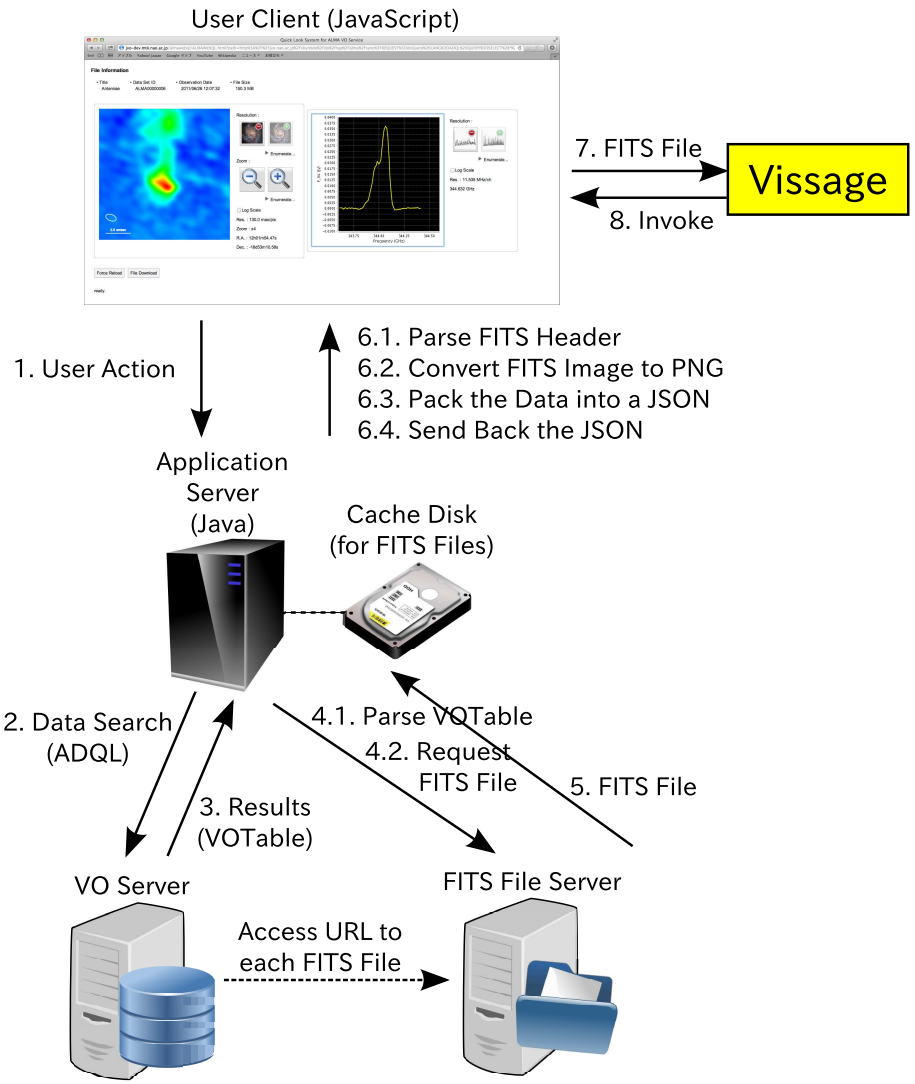
Select spectral range by mouse dragging...



Select spectral range by mouse dragging...



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 applicatin server.

File I/O Speed

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

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

Endian

▷Little Endian

- 0x12345678 → 78 56 34 12
- ex) Intel, AMD

▷Big Endian

- 0x12345678 → 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
  $\rightarrow$ on ahead endian conversion method
- Perform endian conversion and compute the summation at the same time in one loop
  $\rightarrow$ 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

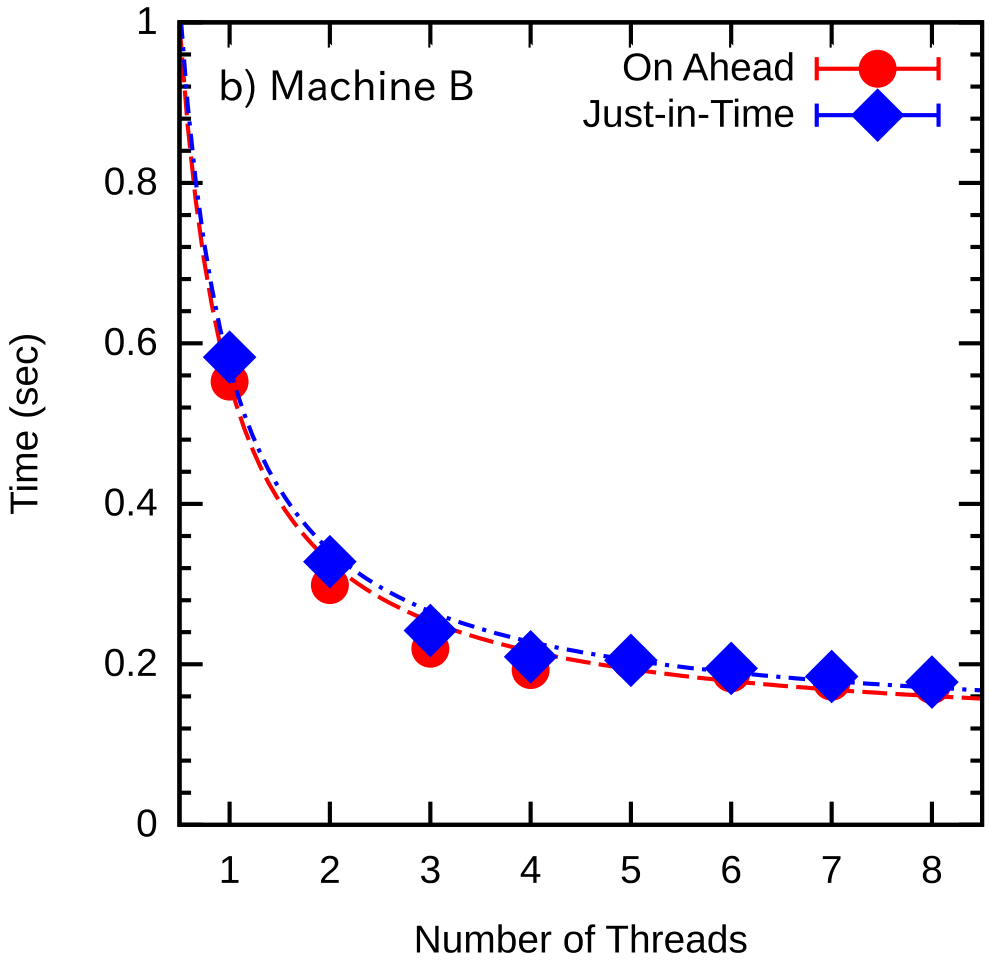
```
{
  for (size_t i = 0; i < len; ++i) {
    v[i] = convert_endian(v[i]);
  }

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

Just-in-Time Endian Conversion Method

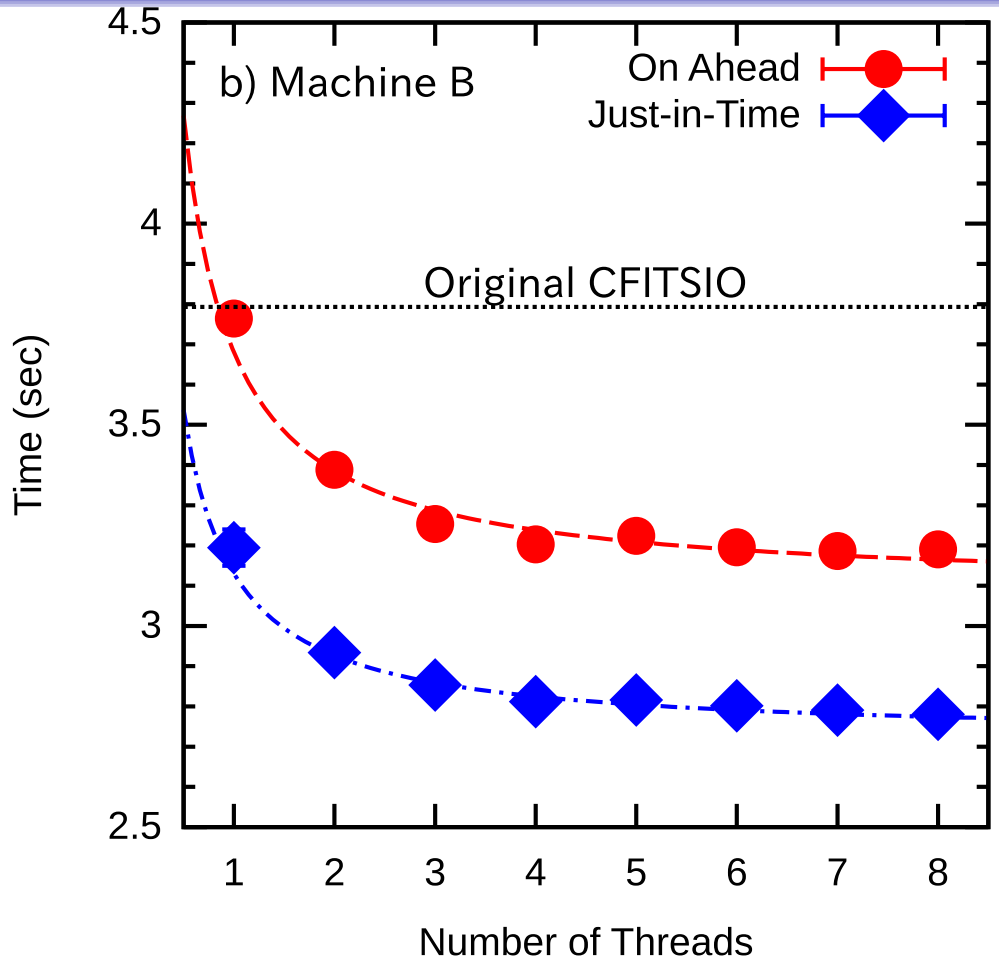
```
{
  double sum = 0.0;
  for (size_t i = 0; i < len; ++i) {
    sum += convert_endian(v[i]);
  }
}
```

Benchmark 2



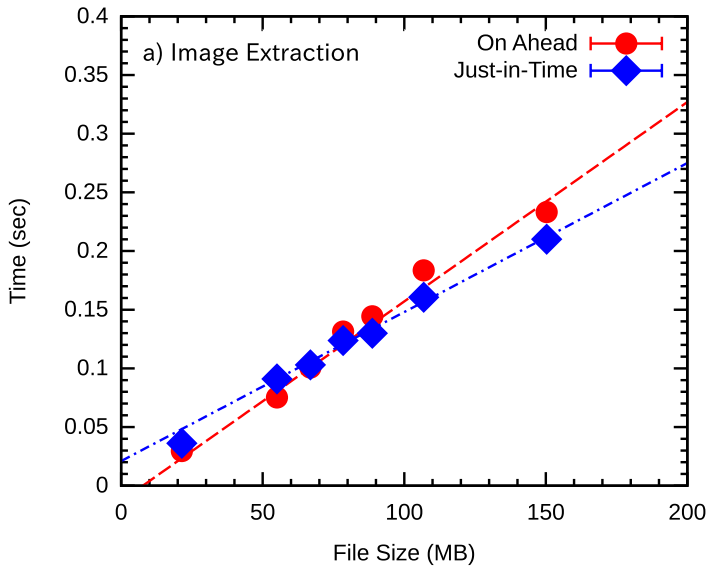
- We observed no significant difference of computation time between both methods (→ zero time endian conversion).

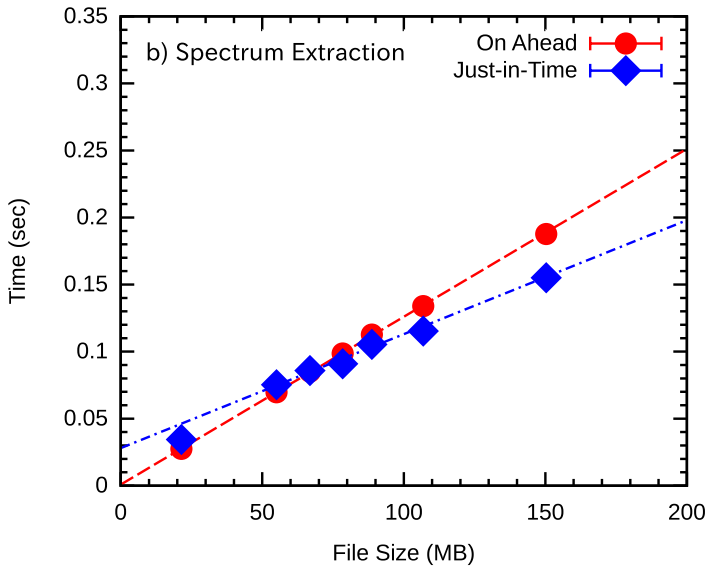
Benchmark 3



- 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.