



ANALYZE IT

User Guide

Copyright

Copyright © 2015-2020, UCit.
All rights reserved.

We Would Like to Hear from You

You can help us make this document better by telling us what you think of the content, organization, and usefulness of the information. If you find an error or just want to make a suggestion for improving this document, please send feedback to UCit Support. Although the information in this document has been carefully reviewed, UCit does not warrant it to be free of errors or omissions. UCit reserves the right to make corrections, updates, revisions, or changes to the information in this document.

UNLESS OTHERWISE EXPRESSLY STATED BY UCit, THE PROGRAM DESCRIBED IN THIS DOCUMENT IS PROVIDED "AS IS" AND WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. IN NO EVENT WILL UCit BE LIABLE TO ANYONE FOR SPECIAL, COLLATERAL, INCIDENTAL, OR CONSEQUENTIAL DAMAGES, INCLUDING WITHOUT LIMITATION ANY LOST PROFITS, DATA, OR SAVINGS, ARISING OUT OF THE USE OF OR INABILITY TO USE THIS PROGRAM.

Document Redistribution and Translation

This document is protected by copyright and you may not redistribute or translate it into another language, in part or in whole, without the express written permission of UCit.

Trademarks

Linux is the registered trademark of Linus Torvalds in the United States and other countries.

Firefox® and Mozilla® are trademarks or registered trademarks of the Mozilla Foundation in the United States and/or other countries.

Apple®, Mac®, Mac® OS X® and Apple® Safari® are trademarks or registered trademarks of Apple, Inc. in the United States and other countries.

Altair® PBS Professional® is a trademark of Altair Engineering, Inc.

SLURM™ is a trademark of SchedMD LLC.

Google™ and Chrome™ are trademarks of Google Inc.

Red Hat® is a trademark of Red Hat, Inc.

Sun® and JavaScript® are registered trademarks of Oracle and/or its affiliates.

Univa® and Univa® Grid Engine® (UGE) are trademarks of Univa Corporation.

Other names mentioned in this document may be trademarks of their respective owners.

Learn about Analyze-IT and UCit products

World Wide Web page

You can find the latest information about UCit Analyze-IT on its web site

<https://www.ucit.fr/analyze-it/>

For more information about other UCit products and about the professional services provided by UCit you can refer to the company's web site <https://www.ucit.fr/>

UCit Analyze-IT Download

The latest Analyze-IT downloadable package and documentation are available at:

<https://www.ucit.fr/download-products>

UCit Support Contacts

Use one of the following to contact UCit technical support.

- Email: helpdesk@ucit.fr
- Helpdesk portal: <https://helpdesk.ucit.fr/>

Table of Contents

| | |
|---|----|
| Copyright | 3 |
| We Would Like to Hear from You | 3 |
| Document Redistribution and Translation | 3 |
| Trademarks | 4 |
| Learn about Analyze-IT and UCit products..... | 4 |
| World Wide Web page | 4 |
| UCit Analyze-IT Download..... | 4 |
| UCit Support Contacts..... | 4 |
| Introduction | 7 |
| How it works..... | 8 |
| Key Features..... | 9 |
| Requirements | 10 |
| Hardware..... | 10 |
| System | 10 |
| Job scheduler | 10 |
| Web browser..... | 11 |
| License..... | 11 |
| Installation..... | 12 |
| Extract Job Scheduler logs | 13 |
| Run Analysis..... | 14 |
| General options | 14 |
| Selecting analysis..... | 15 |
| Concurrent users: -a concUsers | 15 |
| Consumers: -a groupDetails | 15 |
| Congestion: -a congestion | 16 |
| Resubmission of non-completed jobs: -a jobResubmission | 16 |
| Job status: -a jobStatus | 16 |
| Cluster load: -a load | 16 |
| Resources consumption: -a resources | 17 |
| Jobs throughput: -a throughput | 17 |
| Analysis period | 17 |
| Job Scheduler parser | 18 |
| Cluster information..... | 19 |
| Report rendering options | 19 |

| | |
|---|----|
| Creating analysis profiles | 19 |
| Changing colors of graphs..... | 20 |
| Example of usage | 20 |
| Troubleshooting..... | 22 |
| Sometimes, using -a option returns an error | 22 |

Introduction

The job scheduler is the central point for your HPC infrastructure. It dispatches and monitors the jobs on the available resources, keeping track of the jobs' allocation and their state. The generated data are stored in the job scheduler's accounting logs, from which valuable information can be extracted in order to understand the cluster behavior.

For example, the analysis of the logs makes it possible to identify practices and historical events that influence the overall cluster performance: which users are requesting more resources; which kind of jobs wait longer in the queue before starting; resources consumption per job state, and so on. Understanding the historical behavior of the cluster is crucial to choose the proper actions towards increasing production and profitability.

That is why we have developed **Analyze-IT**. By processing the job-submission historical data in the logs, Analyze-IT creates detailed analyses of the behavior of the jobs on your cluster. Moreover, it delivers a report containing indicators and **insights about your HPC infrastructure**.

How it works

The first step is to extract the job submission logs of the cluster. This is done by running the extraction scripts provided with Analyze-IT. This process generates three files with different extensions: `.jobs` (with information about the cluster jobs), `.partitions` (with information about the cluster partitions/queues), and `.nodes` (information about the cluster nodes). They are automatically saved according to the following name format:

```
<cluster_name>_YYYY-MM-DDTHH:MM:SS.<extension>
```

Currently, Analyze-IT uses only the jobs file `<cluster_name>_YYYY-MM-DDTHH:MM:SS.jobs` to carry out the analysis. As depicted in Figure 1, that file is fed to the data-analysis algorithms of Analyze-IT, which calculate a set of metrics defined by the user via input parameters. At the end of the data processing, an html report is generated: it shows the calculated metrics via tables and charts, providing the user with key information about the current state of the cluster and how it has been behaving over time.

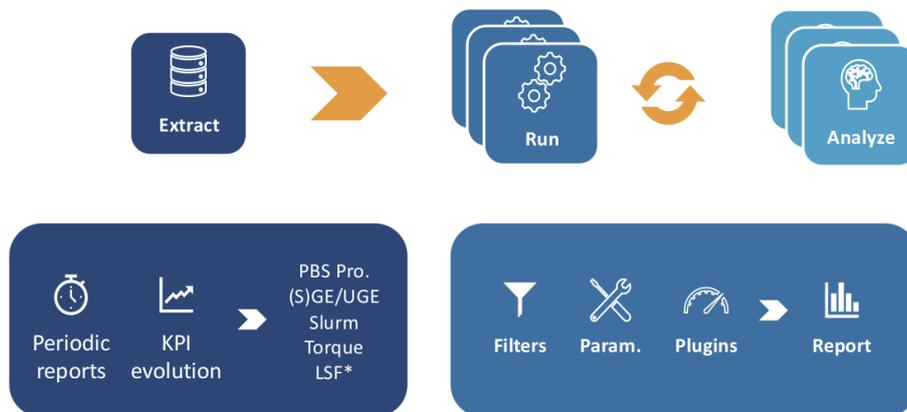


Figure 1. Analyze-IT process

Among the general metrics provided in the report are: global information about the cluster, throughput, concurrent users, resubmission, load, consumers... Each of those has its own page in the report, where several other sub-metrics are provided.

Key Features

You will find below a list of the metrics that can be presented in the report pages. Later in this document, you will discover how you can switch on/off the calculation and display the following metrics:

- Home
 - General information
- Concurrent users
 - who have submitted at least 1 job per [10 min, 1 hour, 1day, 1 month, 1 year]
 - who have at least 1 job waiting or running per [10 min, 1 hour, 1day, 1 month, 1 year]
- Congestion
 - Cluster state (Optimal, Acceptable, Contention, Congestion)
- Consumers by Account, GID, JobName, Partition/Queue, QOS/PE, UID
 - Job count
 - Execution time
 - Consumed core hours
 - Allocated cores
 - Waiting time
 - Slowdown
 - Details for each element in the group (per consumer)
- Job status
 - Percentage of jobs per job status
 - Percentage of core-hours per job status
 - Percentage of jobs per job status each month
 - Consumed resources repartition per job status each month
- Load
 - Cluster load (cores allocated to jobs)
 - Number of jobs per node
- Resources
 - Number of cores and nodes allocated
 - Requested and consumed memory
 - Requested and consumed memory vs. allocated cores
- Resubmission
 - For CANCELLED, TIMEOUT and FAILED jobs
 - Resubmitted job state, parameters
- Throughput
 - Inter-arrival
 - Slowdown
 - Amount of jobs arriving per [10 min, 1 hour, 1day, 1 month, 1 year]
 - Job submission time
 - Job submission weekday

Requirements

Hardware

Analyze-IT should be installed on a virtual machine or physical server separated from the main cluster frontend node. Minimal hardware requirements are:

- CPU: 4 cores
- Memory: 8GB
- Disk space: At least 4GB of free disk space (for tool-generated files and installation files)

Note that the strongest requirement is the RAM: the more jobs you have in your logs, the more RAM you need.

System

- Operating Systems: RHEL 7/8, Centos 7, Ubuntu 16.04/18.04, Debian 8/9/10
- A user account, different from the root account to run Analyze-IT (e.g., `aituser`)

Job scheduler

The following job schedulers are supported:

| Name | Version | Notes |
|---|---|---|
| SLURM™ | 14.11.6 or later | <p>SLURM™ binaries must be installed on the server where you will run the log extraction script, and in the PATH. The <code>sacct</code> command must be usable by the user (e.g., <code>aituser</code>) who runs the script.</p> <p>SLURM™ SlurmDBD server must be reachable from the server where you will run the log extraction script.</p> <p>Accounting must be turned on with SlurmDBD: <code>AccountingStorageType</code> must be set to <code>accounting_storage/slurmdbd</code>, <code>JobAcctGatherType</code> must be set to <code>jobacct_gather/linux</code>¹ (see https://slurm.schedmd.com/accounting.html for more details)</p> |
| Open Grid Engine (GE), Univa® Grid Engine® (UGE) | GE 6.0u6 or later UGE 8.5.4 or later | <p>Relies on the accounting file of Grid Engine. This file only exists if accounting is turned on (<code>accounting=true</code> in the "reporting_params" configuration option). This file must be readable from the server where you will run the log extraction script (default <code>\$SGE_ROOT/\$CELL/common/accounting</code>)</p> |

¹ `jobacct_gather/cgroup` does not gather correctly memory usage in SLURM™ version lower than 17.02

| | | |
|----------------------------------|---------------|--|
| Altair® PBS Professional® | 13.0 or later | The PBS Professional® logging directory must be readable from the server where you will run the log extraction script (default directory is <PBS_HOME>/server_priv/accounting/). |
| Torque | 5.0 or later | The Torque logging directory must be readable from the server where you will run the log extraction script (e.g., <TORQUEROOT>/server_priv/accounting/). |

Web browser

Analyze-IT produces HTML reports that can be viewed with most popular browsers. Generally speaking, Analyze-IT supports the latest versions of each major platform's (Linux®, Apple® Mac® OS X®, and Microsoft® Windows®) default browsers (Google™ Chrome™, Mozilla Firefox®, Apple® Safari®, Microsoft® Edge®). JavaScript® must be enabled on browsers.

License

You need a valid license to install and run Analyze-IT. If you do not have one yet, please contact helpdesk@ucit.fr or your Analyze-IT reseller.

Installation

Analyze-IT is installed via the execution of the file `analyzeit-xx_OS-yy.run` (with `xx` the version of Analyze-IT, `OS` the name of the OS – `rhel`, `ubuntu` or `debian` – and `yy` the version of the OS)

1. Save the `analyzeit-xx_OS-yy.run` file in a folder of your choice, and execute it:
`./analyzeit-xx_OS-yy.run`
2. A welcome message will be printed, and you will be asked if you want to carry on the installation:

```
#####
#                               Welcome to Analyze-IT installer
#####

This installer will guide you during the installation of Analyze-IT.
#####
Press "Enter" to continue or press "q" to quit: █
```

Press "Enter" to continue.

3. The license agreement is shown. To continue with the installation, you should type "accept", otherwise type "quit" to decline (the latter will terminate the installation).
4. Choose the installation path (in this example the installation path is `/home/aituser/analyzeit`) and press "Enter":

```
Enter the path where Analyze-IT must be installed.
#####
Installation directory:
/home/aituser/analyzeit █
```

5. The files will be installed in the chosen folder:

```
Installation of Analyze-IT is now complete
Installation directory: /home/aituser/analyzeit
Log file: /tmp/Analyze-IT-install-1.0-2018-01-22_14:23:29.log
#####
Press "Enter" to exit
```

If you inspect the installation directory, you will find the following structure:

- `X.Y`: directory containing Analyze-IT version `X.Y`
- `bin`: directory containing the binary script that calls the current version of Analyze-IT and the binary script that extracts data from the job scheduler
- `conf`: configuration directory, currently unused
- `current-version`: file containing the current version of Analyze-IT
- `license`: directory containing the license file
- `log`: directory containing the installation logs

The installation can also be performed in "silent" mode if you execute the following command:

```
./analyzeit-xx_OS-yy.run --accept -- -b -i "installation_dir_path"
```

Extract Job Scheduler logs

Analyze-IT provides a script to extract the historical data from the job scheduler. You can find it in `<INSTALLATION_PATH>/bin/extractData`. The first argument is the name of the job scheduler, subsequent arguments may vary depending on the job scheduler (type `<INSTALLATION_PATH>/bin/extractData JOBSCHED -h`, with `JOBSCHED: pbs, torque, slurm` or `sge`). The data extraction relies on different methods depending on the job scheduler:

- SLURM™: relies only on `sacct` and `scontrol`.
- Torque and Altair® PBS Professional®: extracts raw data from accounting files, and from `pbsnodes` and `qstat`.
- Grid Engine: extracts raw data from accounting file

The script creates 3 text files in the current directory:

- **historical data on jobs:** `HOSTNAME_DATE.jobs`
 - From a given date up to now
 - Retrieve all data gathered by the job scheduler (job id, requested/obtained resources, node list, submit/start/run times...)
 - For confidentiality reasons, the output is anonymized: by default, usernames and groups are not retrieved, only UID and GID are
- current **list of nodes** and their description: `HOSTNAME_DATE.nodes`
 - Node name
 - Number of cores, number of cores per socket, memory
 - Specific resources (features described in the job scheduler)
- current **list of partitions/queues** and their description: `HOSTNAME_DATE.partitions`

Currently, Analyze-IT only relies on the first file: `HOSTNAME_DATE.jobs`.

The execution of the script is very lightweight. No processing of the data is done.

`extractData` can also retrieve the data through from a remote server through ssh: simply specify the `-h` hostname, `-u` user and `-i` ssh-key options before the name of the job scheduler to do so.

`./bin/extractData <-s hostname> <-u user> <-i ssh-key> [pbs|sge|slurm|torque] <options>`

Run Analysis

Navigate to the folder where Analyze-IT was installed

```
cd <INSTALLATION_PATH>
```

Analyze-IT is run via the execution of the file `bin/analyzeit`. If you call it with the help option `-h` or `--help`, it lists all possible analyzes and options:

```
bin/analyzeit --help
```

General options

We describe below the analyzes you have access to; with the associated options you can use to customize any analysis to your needs. On top of all these options, you also have two general options for running Analyze-IT:

- `-v, --version`: Show program's version number and exit
- `-o OUTPUT, --output OUTPUT`: Output directory for the report (default: report)
- `-mnp MAXNBPROCESSES, --maxnbprocesses MAXNBPROCESSES`: Maximum number of processes to launch for the analysis (default: 8)
- `-nw NBWRITERS, --nbwriters NBWRITERS`: Number of writer processes (default: 2)

Data enhancers

Analyze-IT offers a couple of options for you to consolidate the data retrieved within the job scheduler logs with external sources of data:

- `-de, --dataenhancers`: If true, then apply all data enhancers found in `AIT_CONF_PATH/enhancers` directory (default: False)
- `-def DATA_ENHANCERS_FILL_VALUE, --dataenhancersfillvalue DATA_ENHANCERS_FILL_VALUE`: String to fill in missing values in data enhancers. If not specified, values might be ignored by the analysis. (default: None)

Data Enhancers must be implemented in either csv (comma separated value) or python files in the `AIT_CONF_PATH/enhancers` directory:

1. CSV (.csv) files: These are the simplest and safest enhancers. Each csv file can add multiple features (columns) to the data processed by Analyze-IT, each csv file must contain:
 - a. on the first line the name of the columns: the first one is the pivot column (and must exist in the input Data – see the `-fc` option for a list of available columns), the other ones are the names of the new columns that will be created (Analyze-IT will add a `'_'` prefix to all the column names specified in the csv file).
 - b. following lines must contain the values: first one is the pivot value, next ones are the newly created values
 - c. format of the csv file is "free" for as long as the Analyze-IT is capable of detecting the format (you can use comas, semi-colons... as separators)
2. python (.py) files: These Data Enhancers allow you more complex and dynamic data transformations, as it you can either add new columns, or modify values in existing ones. All new column names must be prefixed by `'_'`. Python Data Enhancers receives a Python

DataFrame² representing the data processed by Analyze-IT. The class implemented in the Python file must respect the following conventions:

- a. the name of the class must be the same as the name of the file without the .py extension, and with an upper case first. For example, if your file is `myDataEnhancer.py`, then the class must be `MyDataEnhancer`.
- b. the `__init__` method receives a single argument: `prefix`, the prefix that needs to be prepended to any new column name.
- c. a `transform(self, dataframe)` method must be implemented. The modifications on the DataFrame must be made in place. No return value is expected.
- d. a `get_created_columns(self)` method must be implemented. It must return a list of strings representing the names of the newly created columns.

All data added or modified by Data Enhancers can then be queried when using the `-fc` option and will be analyzed by the `groupDetails` analysis. Note that all features are internally added with a `'_'` prefix, so if you add a feature named `FEAT`, you will need to query it with `__FEAT`.

Examples of Data Enhancers are presented in annex at the end of this document.

Selecting analysis

By default, Analyze-IT runs all the available analyses it offers, but you can also specify which ones you would like to see in your final report (with the `-a` option). Note that the order with which you specify the list of analysis you want to run has an impact on the way the analysis will be displayed in the generated report: the links will follow the order you specified.

Concurrent users: `-a concUsers`

Computes how many different users are using the cluster at the same time.

Associated options:

- `-cr {10min,1hour,1day,1week,1month,1year} [{10min,1hour,1day,1week,1month,1year} ...], --concusersres {10min,1hour,1day,1week,1month,1year} [{10min,1hour,1day,1week,1month,1year} ...]`: Time resolution for analysis of concurrent users. (default: '1hour', '1day', '1month')

Consumers: `-a groupDetails`

It groups the jobs on similar JobName, UID, GID, Partition, Account, or QOS, and displays statistical information about these groups

Associated options:

- `-g CONSUMERS_GROUP [CONSUMERS_GROUP ...], --group CONSUMERS_GROUP [CONSUMERS_GROUP ...]`: Name of the columns to group the data with to analyze consumption. (Note: `'_*'` refers to all columns created by data enhancers. You can also

² <https://pandas.pydata.org/pandas-docs/stable/reference/api/pandas.DataFrame.html>

specify only the ones you want - wrong columns will be silently ignored) (default:

`['JobName', 'UID', 'GID', 'Partition', 'Account', 'QOS', 'WCKey', '_*']`)

- `-gj, --groupjobnames`: Group similar job names using fuzzy string matching: similar job names will be replaced by a single job name that is considered as the “main” name. For example, if Analyze-IT has detected that the following job names are similar [“job1”, “job2”, “job3”...] and that “job1” is the most significant name, all jobs in the group will have their job name replaced by “job1”. Warning: using this option consumes a lot of time and RAM.
- `-gds GROUPDETAILNBSAMPLES, --groupdetailnbsamples GROUPDETAILNBSAMPLES`: Maximum number of points (jobs) on the graphs on the details analysis pages. If None, all jobs are displayed. (default: 5000)
- `-rn, --realnames`: Use real user and group names if available instead of UID/GID. (default: False)

Congestion: **-a congestion**

Cluster state (Optimal, Acceptable, Contention, Congestion) through time, and jobs life cycle

Associated options:

- `-rr, --run_ratio`: Ratio of full cluster usage defined as optimal limit (default: 0.8).
- `-wr, --wait_ratio`: Ratio to specify what is considered as acceptable time to wait compared to running time. (default: 2)
- `-cor {1day,1week,1month} [{1day,1week,1month} ...], --congestion_res {1day,1week,1month} [{1day,1week,1month} ...]`: Time resolution for analysis of core running and waiting time (default: '1day', '1week', '1month')

Resubmission of non-completed jobs: **-a jobResubmission**

It tries to detect if failed, cancelled or timeout jobs have been resubmitted.

Associated options:

- `-msr MAXSECONDSRESUBMIT, --maxsecondsresubmit MAXSECONDSRESUBMIT`: Maximum of seconds allowed between two jobs to consider a job resubmission (default: 172800, i.e., 48 hours)

Job status: **-a jobStatus**

It displays the repartition of number of jobs and consumed cpu-hours per job state, globally and per month.

Cluster load: **-a load**

It displays the load of the cluster in terms of number of allocated cores and the number of jobs allocated to nodes.

Associated options:

- `-lt {cluster,nodes} [{cluster,nodes} ...], --loadtypes {cluster,nodes} [{cluster,nodes} ...]`: Types of load to be analyzed. (default: 'cluster', 'nodes')

- `-cls CLOADNBSAMPLES, --cloadnbsamples CLOADNBSAMPLES`: Maximum number of points (jobs) on the graphs on the cluster load analysis page. If 0, display all points. (default: 10000)

Resources consumption: **-a resources**

Computes the resources consumption: cores, memory, node...

Associated options:

- `-r {core,memory,node} [{core,memory,node} ...], --resources {core,memory,node} [{core,memory,node} ...]`: Resources consumption to be analyzed. (default: 'core', 'memory', 'node')
- `-cg [CORESGROUPS [CORESGROUPS ...]], --coresgroups [CORESGROUPS [CORESGROUPS ...]]`: List of number of cores: used to group jobs by number of allocated cores

Jobs throughput: **-a throughput**

It analyzes the frequency at which jobs are submitted and enqueued by the job scheduler, and their slowdown.

Associated options:

- `-tr {10min,1hour,1day,1week,1month,1year} [{10min,1hour,1day,1week,1month,1year} ...], --throughputres {10min,1hour,1day,1week,1month,1year} [{10min,1hour,1day,1week,1month,1year} ...]`: Time resolution for analysis of job throughput. (default: '10min', '1hour', '1day', '1week', '1month', '1year')

Analysis period

The following options allow you to consider only jobs in a given time frame, both can be used at the same time:

- `-s "%Y-%m-%d[%H:%M:%S]", --starttime "%Y-%m-%d[%H:%M:%S]"`: Consider jobs that were submitted after this date (default: "1970-01-01 00:00:00")
- `-e "%Y-%m-%d[%H:%M:%S]", --endtime "%Y-%m-%d[%H:%M:%S]"`: Consider jobs that were submitted before this date (default is the current date)

Start and end dates (`-s` and `-e` options) can be specified without any hour. In this case the start of the day (00:00:00) is used for the start date, and the end of the day (23:59:59) for the end date.

Job Scheduler parser

Analyze-IT does not detect automatically which job scheduler you took your data from. You must specify the right parser Analyze-IT needs to use. The following options allow you to configure this:

- `-j {slurm,torque,pbs,sge}, --jobscheduler {slurm,pbs,torque,sge}`: Job scheduler parser to use (default: `slurm`).
- `-jp JSPARAMS, --jparams JSPARAMS`: Job scheduler additional parameter. Format is "key=value, key=value...".
 - **Grid Engine**
 - `acct_file`: path to accounting directory (default: `/usr/share/gridengine/default/common/accounting`)
 - **Altair® PBS Professional®**:
 - `acct_dir`: path to accounting directory (default: `/var/spool/pbs/server_priv/accounting/`)
 - **Slurm**:
 - `noalloc`: Gather details about each job step or not? If `noalloc` is `True` (default), then we gather the details for all steps of the jobs, allows to get the `MaxRSS` values properly.
 - **Torque**:
 - `acct_dir`: path to accounting directory (default: `/var/spool/torque/server_priv/accounting/`)
- `-sd, --savedata`: Save data once parsed into a `.pickle` file. Allows for faster further analyses. (default: `False`). The file will be saved in the same directory as the input file, with the same name, and a `.pickle` extension. You can then use directly provide the `.pickle` file to Analyze-IT as an input file. Note that if you specify a start or end time (`-s` and `-e` options), only jobs corresponding to the selected period will be saved in the `.pickle` file.
- `-sfd, --savefiltereddata`: Save data once filtered (see `-fc` option) into a `.filtered.pickle` file. Allows for faster further analyses. (default: `False`). You can then use directly provide the `.filtered.pickle` file to Analyze-IT as an input file.
- `-fc FILTERCOLUMN, --filtercolumn FILTERCOLUMN`: Filter columns to keep only certain jobs. When analyzing your job scheduler data, you may want to analyze only a subset of the data depending on the value of certain fields, for example you may want to analyze the behavior of certain groups of users, and thus only keep jobs submitted by a list of GIDs for the analysis and drop all the other jobs. Analyze-IT allows you to filter the input data to really target what you want to analyze.
 - Format is the following: `columnName<FILTER>value` or `columnName<FILTER>value1,value2,...`
 - Multiple filters can be provided by separating them with `'`.
 - FILTER can be: `==, !=, <=, >=, <, >, ~=` (regex). Example: `UID==1234;JobName!=Abaqus,Fluent...`
 - Available columns (depends on the input data) are: `Account, Allocated_CPUS, Allocated_Nodes, _Memory, Cluster, Comment, Eligible, End, Exit_Code, GID, Group, JobID, JobName, MaxRSS, NodeList, Partition, QOS, Requested_CPUS, _Nodes, Requested_Memory, Requested_Memory_Per_CPU, Start, State, Submit, Timelimit, UID, User`

Cluster information

Sometimes the input data do not contain all the necessary information to really compute the actual values of some analysis. The following options allow you to set default values when they are missing in the input data (you will usually see 'nan' showing up in the result in the analysis when some values are missing):

- `-c CORESPERNODE, --corespernode CORESPERNODE`: Number of cores per node. Used to compute the number of cores used by a job when only the allocated number of nodes is known in the logs, and vice-versa. (default: 1)
- `-tc TOTALNBCORES, --totalnbccores TOTALNBCORES`: Total number of cores on the cluster per period. Format is the following: a string "nbcores[, YYYY-MM-DD:nbcores...]" (the date represents the start date at which point the number of cores change and take a new value). (default: None)
- `-dq DEFAULTQUEUE, --defaultqueue DEFAULTQUEUE`: Default queue name to use when queue is absent in the input data. (default: None)

Report rendering options

The following options allow you to customize the report (add a logo, change its name, modify the graphs...):

- `-n name, --name name`: Name of the analysis. By default, this will be the name of the input file. (default: None)
- `-l LOGO, --logo LOGO`: Path to PNG logo to display on the top of the pages (recommended size: height="80" width="160") (default: None)
- `-op OPACITY, --opacity OPACITY`: Opacity of RGB color for plotting. Value should be a real number in [0, 1] (default: 0.2)
- `-nm, --nominification`: Do not minify HTML files. By default, HTML files are minified to save space, which make them less human readable. (default: True)

Creating analysis profiles

If you often use the same options to run your analysis, you can create and reuse profiles that contain selected sets of options.

Profiles are saved in Analyze-IT configuration folder (`conf` in the installation directory) in files named `profile_name.profile`. These files contain on each line an option for Analyze-IT.

Every time you run Analyze-IT, a `last_run.profile` file is generated that contains all the options you used for this analysis. To create your own profiles, you can either rename this file to save it and reuse it in subsequent executions of Analyze-IT, or you can run Analyze-IT with `@myprofile`:

1. If `conf/myprofile.profile` exists, then Analyze-IT will load all the options stored in this profile and use them.
2. If `conf/myprofile.profile` does not exist, it will be created and used options will be saved in this profile

Note that you can overload the options present in a profile simply by specifying them on the command line: the command line options take precedence over the profile options.

Changing colors of graphs

Update the `rgb_colors.csv` file in the `template` directory, under the `X.Y` directory.

Make sure that the following 'colors' are defined: `CANCELLED`, `COMPLETED`, `FAILED`, `NODE_FAIL`, `PREEMPTED`, `TIMEOUT`, `BOOT_FAIL`, `REQUEUED`, `Unused resources`.

Also make sure that you have at least 6 more colors (hence a total of 15 colors).

Colors are taken in the order they are defined in the file to generate the graphs.

Example of usage

Here it is shown an example of how to run Analyze-IT with its default settings, i.e., performing all the available analysis, and provide an overview of the resulting report.

Navigate to the folder where Analyze-IT was installed

```
cd <INSTALLATION_PATH>
```

To start the analysis, type the following command (It is assumed that the job-scheduler logs are already extracted – check Section “Extract Job Scheduler logs”)

```
./bin/analyzeit -o /tmp/report HOSTNAME_DATE.jobs
```

This will run all the available analysis, with their default values, for the file `HOSTNAME_DATE.jobs`. A web-based report will be generated and saved in the folder `/tmp/report`. Go inside it and open `index.html` with your favorite web browser.

Figure 2 shows an example of the report homepage.

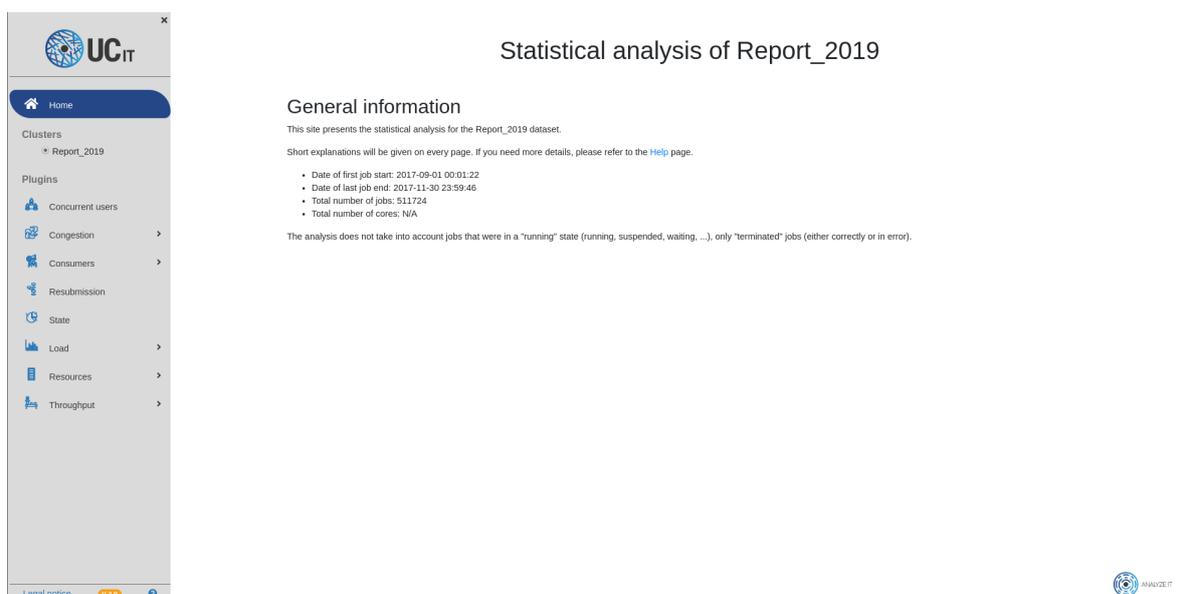


Figure 2. Report homepage

Notice that for each analysis type a navigation link is created. For example, a section of the "State" page is shown on Figure 3, with metrics like the "percentage of jobs per job status" and "percentage of core-hours per job status".

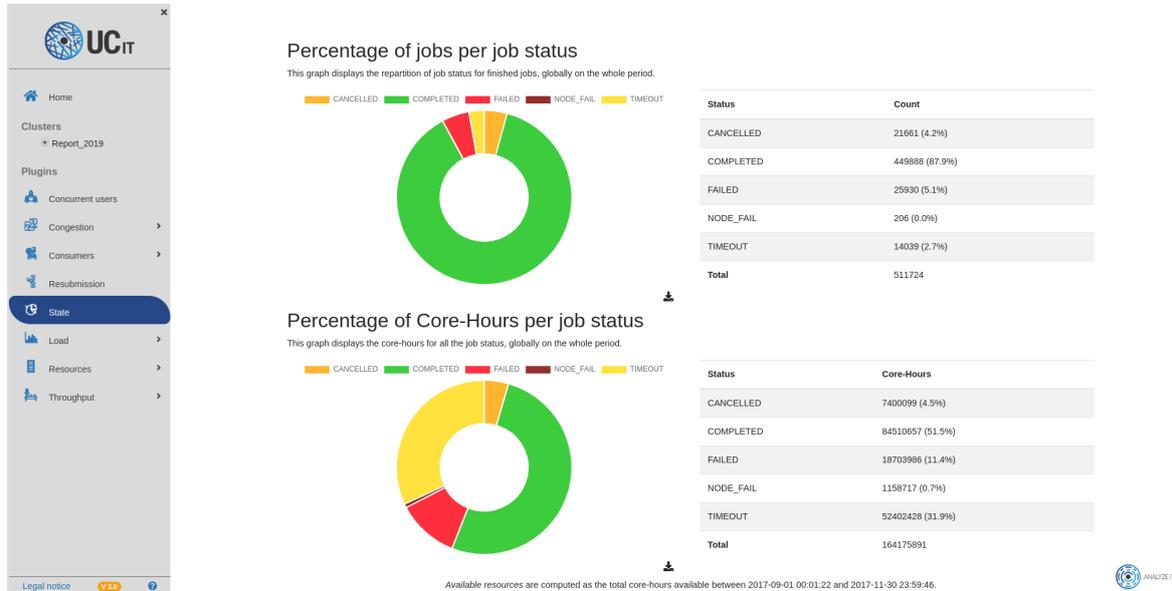


Figure 3. Job status

Another example is shown on Figure 4, the "Consumers" pages shows the Accounts, GIDs, JobNames, Partitions, QOS, and UIDs in terms of Allocated_CPUs, CPU_hours, Execution_Time, Job_Count, and Waiting_Time. The snapshot below (Figure 4) lists the 10 UIDs that consume more CPU_hours.

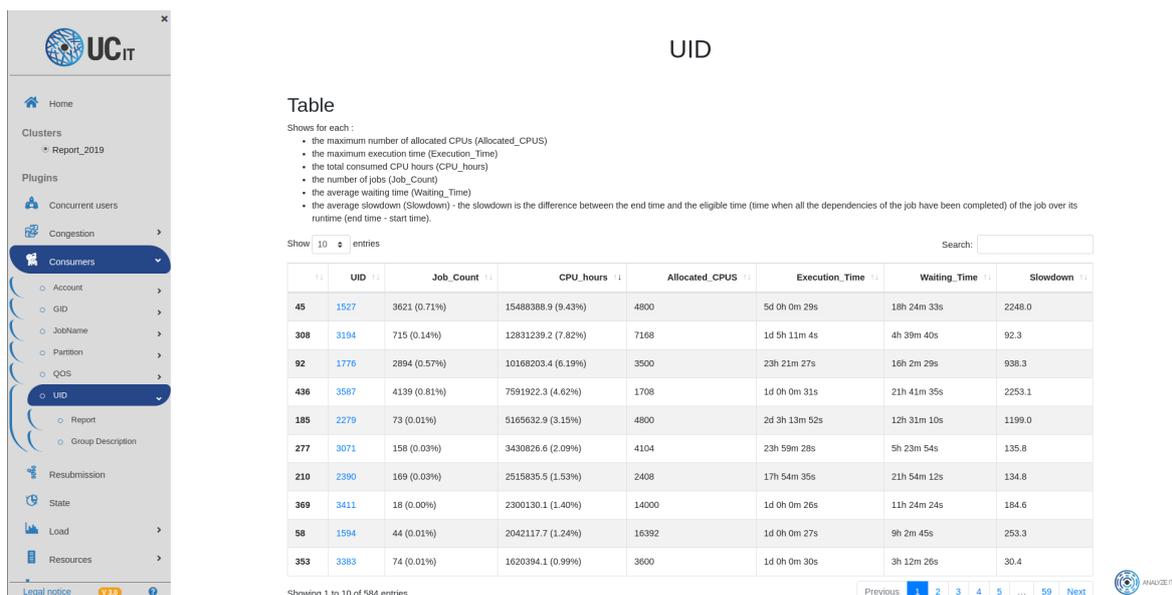


Figure 4. Details of cpu-hours consumption per UID

Troubleshooting

Sometimes, using -a option returns an error

When you want to perform a specific analysis with a subset of features, you must provide the executable of Analyze-IT with the list of requested features using “-a” option.

In this case, your command may produce the standard Usage information followed by the following error:

```
Analyze-IT: error: argument -a/--analysis: invalid choice:
'/PATH/TO/YOUR/INPUTFILE' (choose from 'concUsers', 'groupDetails',
'jobResubmission', 'jobStatus', 'load', 'resources', 'throughput',
'congestion'congestion)
```

In such a case, you can:

- Use the “-a” option between 2 other options in the command line
- Put the “-a” option at the end of the command line, after the path to the `inputfile`
- Use “--” to separate the “-a” option features list and the path to the input file (if “-a” is the last option provided in the command line before the input file)

ANNEX

Data Enhancer – CSV example

The following example creates two new columns (`FirstName` and `Name`) from the `UID` of the user.

```
UID, FirstName, Name
10001, Luke, Skywalker
10002, Anakin, Skywalker
0, Yoda, Yoda
22, R2, D2
```

Data Enhancer – Python example

The following example creates a new column (`_category`) and modifies the `Account` of the jobs.

```
class TestTransform():
    def __init__(self, prefix):
        self.new_cols = []
        self.prefix = prefix

    def transform(self, dataframe):
        # Modifications must be done in place
        # Creation of a new column
        newcol = self.prefix + "category"
        self.new_cols.append(newcol)
        dataframe[newcol] = dataframe["Allocated_CPUS"].apply(lambda x:
"mono" if x == 1 else "multi")

        # Modification of an existing column
        dataframe["Account"] = "Grouped accounts"

    def get_created_columns(self):
        return self.new_cols
```