AlgoRun Documentation

Release 2.0

Abdelrahman Hosny

Mar 09, 2021

Contents

AlgoRun is a docker-based software container template designed to package computational algorithms. These pages show steps of how to create an AlgoRun container of an implemented algorithm.

CHAPTER 1

Packaged Algorithms with AlgoRun before?

Don't forget to use **docker pull algorun/algorun** before you start packaging algorithms, in order to get the latest update of AlgoRun Docker image.

CHAPTER 2

Contents

2.1 Packaging Algorithms

AlgoRun is a docker-based software container template designed to package computational algorithms. These pages show steps of how to create an AlgoRun container of your implemented algorithm.

2.1.1 Packaged Algorithms with AlgoRun before?

Don't forget to use **docker pull algorun/algorun** before you start packaging algorithms, in order to get the latest update of AlgoRun Docker image.

2.1.2 1. Download and Install Docker

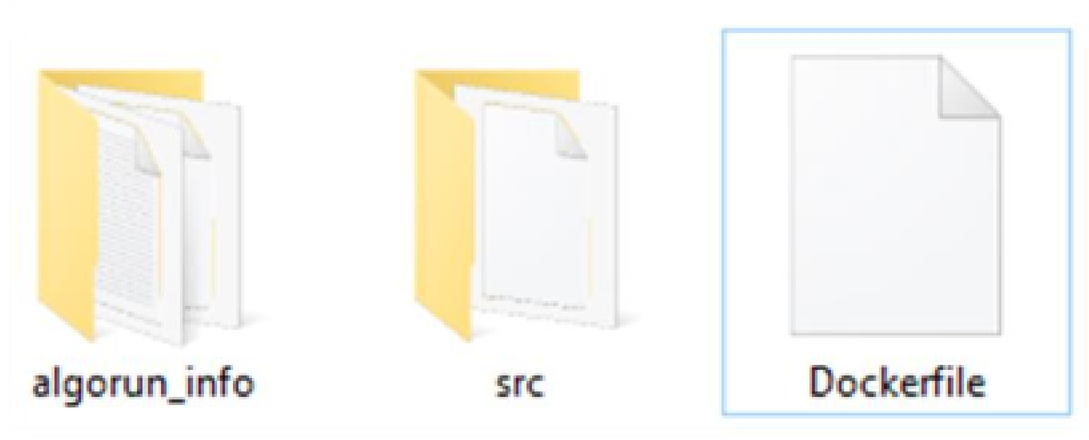
Before starting, download and install Docker on your local machine. Docker can be installed on Mac OS, Linux as well as Windows. Follow the instructions on: https://docs.docker.com/v1.8/installation/

2.1.3 2. Download AlgoRun

Download AlgoRun from: https://github.com/algorun/skeleton

2.1.4 3. Create an AlgoRun-based Docker Image

Unzip the downloaded skeleton-master.zip file. The resulting folder has the following structure:



- A Dockerfile is a text document that contains all commands needed to build the software container. Docker builds a software container automatically by reading the instructions from the Dockerfile.
- AlgoRun template container uses the *src* and *algorun_info* folders to deposit all your source code and to describe the implemented algorithm in a standard format.

STEP 1: Add all source code files of your algorithms in to the *src* folder

STEP 2: Edit the *Dockerfile* to make sure your algorithm dependencies will get installed in the container. AlgoRun is based on Ubuntu 15.10 Linux system so you can leverage Ubuntu packaging system to get your dependencies installed

For more information about Dockerfile, please refer to the Docker documentation (https://docs.docker.com/v1.8/reference/builder/)

STEP 3: Edit the *manifest.json* file inside the *algorun_info* folder. Below is an example of the manifest file.

```
{
        "manifest_version": "2.0",
        "algo_name": "BLAST",
        "algo_version": "2.3.0",
        "algo_summary": "Compares a nucleotide query sequence against a nucleotide sequence database",
        "algo_description": "BLAST for Basic Local Alignment Search Tool is an algorithm for comparing primary biological seque
        "algo_website": "http://blast.ncbi.nlm.nih.gov/Blast.cgi",
        "algo_keywords": ["similarity", "biological", "sequences"],
        "published_paper": {
            "title": "Basic local alignment search tool",
            "url": "http://www.sciencedirect.com/science/article/pii/S0022283605803602"
        },
        "algo_authors": [
                {
                        "name": "Stephen Frank",
                        "email": "",
            "profile_picture": "stephan.jpeg",
                        "personal_website": "https://scholar.google.com/citations?user=VRccPlQAAAAJ&hl=en&oi=sra",
                        "organization": "NCBI, NLM, NIH",
                        "org_website": "http://www.nih.gov/"
                }
        ],
        "algo_packager": {
            "name": "Abdelrahman Hosny",
            "email": "abdelrahman.hosny@hotmail.com",
            "personal_website": "http://www.abdelrahmanhosny.me",
            "profile_picture": "",
            "organization": "University of Connecticut",
            "org_website": "http://cse.uconn.edu"
    },
        "algo_exec": "sh run_blast.sh $input",
        "algo_input": [
                { "name": "input", "src": "file", "type": "text/plain", "accepted_format": "FASTA" }
        ],
        "algo_output": [
                { "name": "output", "src": "output.txt", "type": "text/plain", "format": "BLAST" }
        ],
        "algo_parameters": {
                "caching": "on"
        },
        "algo_image": "algorun:blast"
}
```

In addition to adding algorithm's information, the following fields are necessary for AlgoRun to correctly execute the algorithm source code:

- "algo_exec": is the command used to start algorithm executed.
- "algo_input": is how the algorithm reads the input data.
- "algo_output": is the path of the file where the algorithm outputs its result or stdout if the algorithm prints the result to the standard output stream.

Command line options can be exposed in the "algo_parameters" field (Refer to examples tab for a detailed example using parameters). AlgoRun website uses "input_type" and "output_type" to easily identify algorithms that can communicate together. Please refer to http://algorun.org/input-output-types to see what input and output types you should use. Users can also download and use the algorithm Docker image locally from Docker Hub if the value "algo_image" is provided.

STEP 4: Provide input and output examples in the input_example and output_example folders respectively.

STEP 5: Build the algorithm container from the command line using docker build command: *docker build -t <algorithm_name> .*

2.1.5 4. Run the algorithm

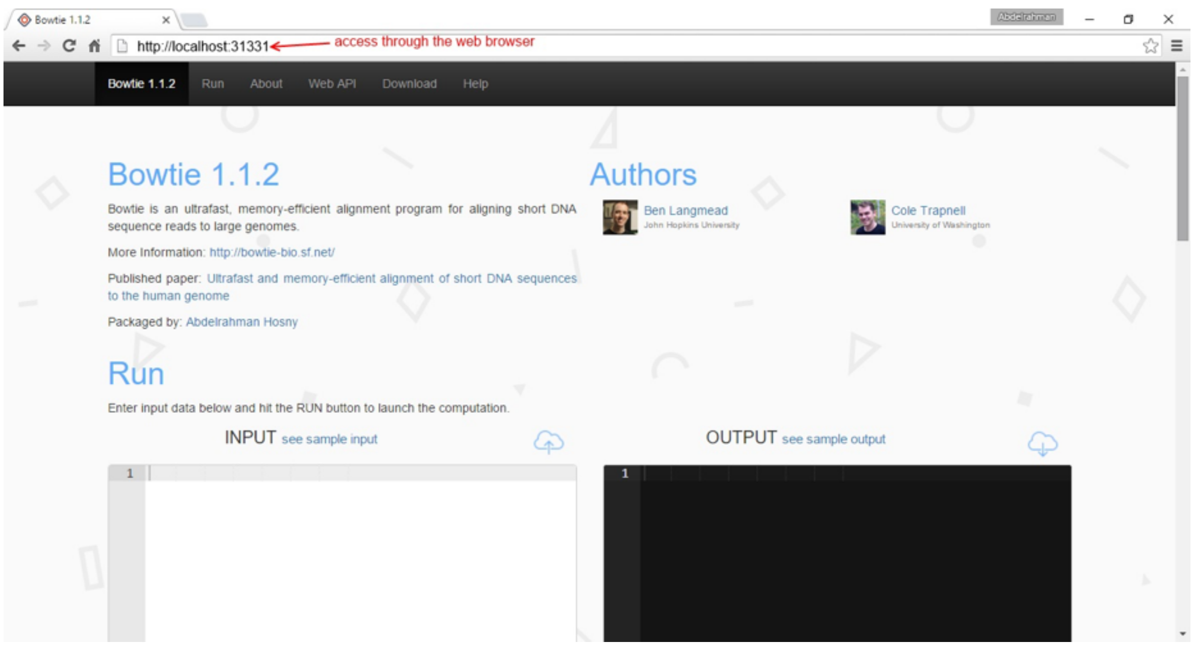
Once the algorithm container has been created, it must first be deployed before the user can start using the packaged algorithm. The container can be deployed with the following command:

docker run -p 31331:8765 –name <container_name> <algorithm_name>

Now that the algorithm container has been deployed on your local machine (localhost), AlgoRun provides the user with three different ways to run the algorithm.

4.1 Web User Interface

The easiest and quickest way to run the packaged algorithm is to open the web browser and type http://localhost:31331

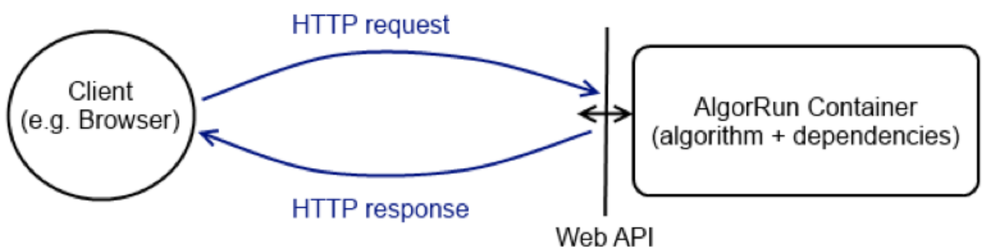


The web user interface of an algorithm packaged with AlgoRun. Type in the address http://localhost:31331 in a web browser to open the web page of the running algorithm container.

4.2 Web API

A web API is an Application Programming Interface (API) used to offer programmatic access to remote resources or services (in our case "computations from an algorithm") that can be accessed by clients such as web browsers or any http-enabled third-party applications. AlgoRun containers are pre-included with a RESTful API[1] that allows access to the computation through the traditional HTTP POST request. Clients communicate to web APIs through a request/response protocol. To ask the web service to perform a computation, a client sends an HTTP request. The body of the request includes necessary input data for the algorithm behind the web service to start. The response of the web service includes the result of the computation.

[1] REpresentational State Transfer (REST) APIs uses Hyper Text Transfer Protocol (HTTP) requests as the main scheme of communication.



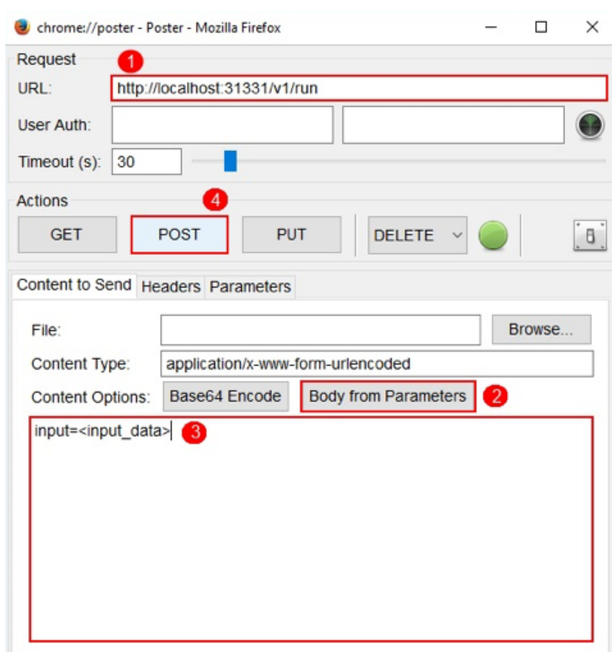
AlgoRun Web API Communication Scheme.

The HTTP request should be sent to a specific address, which is called an endpoint. The two main endpoints exposed by AlgoRun containers are shown below.

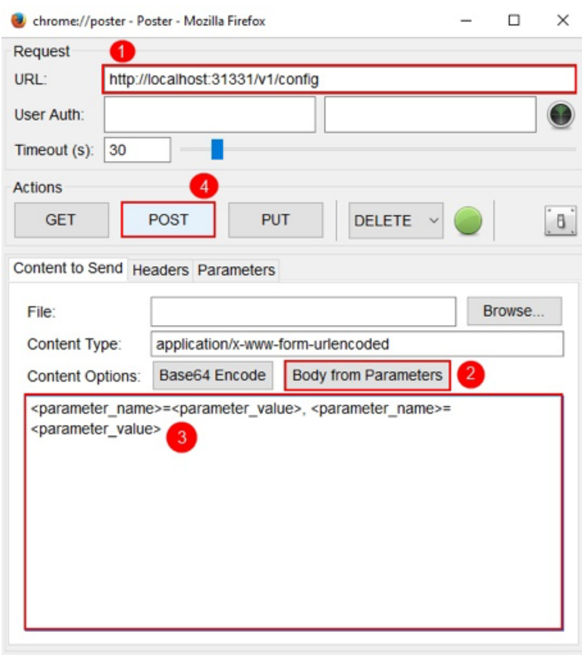
Endpoint	Usage	Reuqest Parameters	Response
HTTP POST /v1/run	used to run the algorithm on a given input data	input: <input_data>	The result of the computation in the body of the response
HTTP POST /v1/config	used to dynamically change the parameters values	<parameter_name>: <parameter_value>	The result of changing the parameter value

By offering standardized input and output, Web APIs are particularly useful when it comes to building complex software applications as they make it easy to integrate different algorithms that usually run on different programming environments. It enhances the modularity of the software, hence increases its robustness and makes troubleshooting problems easier. You can embed computations in large software programs in just a few lines of code, removing the hassle of installing the whole algorithm environment locally.

To test a web API, Firefox Poster Plugin is a graphical user interface tool used to easily send and troubleshoot HTTP requests. See below for examples on using the above AlgoRun endpoints.



Calling Web API using Firefox Poster Plugin: (1) Type the URL of the endpoint http://localhost:31331/v1/run (2) Select Body from Parameters. (3) Type input=<paste_your_input_data_here>. (4) Click Post to initiate the request



Configuring Parameters using Web API: (1) Type the URL of the endpoint http://localhost:31331/v1/config (2) Select Body from Parameters. (3) Type <parameter_name>=<parameter_value>. (4) Click Post to initiate the request.

4.3 Command Line

The traditional command line execution is still available as well.

```
docker exec -i <container_name> /bin/algorun < sample_input.txt
```

2.1.6 5. Publish your algorithm to the AlgoRun website

If you packaged your algorithm with AlgoRun and want to give your algorithm more visibility, we encourage you to submit it for listing on the AlgoRun website. The AlgoRun website serves as a repository for all computational algorithms that were packaged using AlgoRun: http://algorun.org

To submit your algorithm for listing, fill the form located at http://algorun.org/submit-algorithm

2.2 Examples

In this section, we show the process of creating AlgoRun containers for 3 different examples of published software:[1] the popular bioinformatics software Bowtie (Langmead, 2009),[2] REACT (Vera-Licona, 2014), a systems biology software to infer gene regulatory networks and,[3] the KS algorithm to solve the transversal hypergraph generation problem (Kavvadias, 2005). For the first example, Bowtie, we show how to create an AlgoRun container, how to run the Bowtie AlgoRun container using AlgoRun web interface, how to expose command line options as parameters,

[1] Langmead, B. et al. (2009). Ultrafast and memory-efficient alignment of short DNA sequences to the human genome. Genome Biology, 10:R25.

[2] Vera-Licona, P., Jarrah, A.S., Garcia, LD., Mcgee, J., Laubenbacher, R. (2014): An Algebra-Based Method for Inferring Gene Regulatory Networks. BMC Systems Biology, 8:37.

[3] Kavvadias D. and Stavropoulos E. (2005): An Efficient Algorithm for the Transversal Hypergraph Generation. J. of Graph Alg & App, 9:2, 239-264.

some input examples to highlight the use of command line options vs. parameters and finally, how to access the tool deployed with AlgoRun via a RESTful API interface. For the other two examples we show how to create the AlgoRun containers and provide the users with the appropriate links to allow users to deploy and use all the AlgoRun features as presented in the first example.

2.2.1 Packaging Bowtie Software with AlgoRun

Bowtie (Langmead, 2009) is an ultra-fast memory-efficient short read aligner. The source code is written in C++ and is available under the Artistic License. Download it from http://sourceforge.net/projects/bowtie-bio/files/bowtie/1.1.2/

Unzip the downloaded file. This unzipped file will contain all the source code of Bowtie.

STEP 1: Add all Bowtie source files inside the src folder.

STEP 2: Add the instructions to install the C++ dependencies as well as the instructions to build Bowtie source code to the Dockerfile. Below is how the Dockerfile of Bowtie looks like.

```
FROM algorun/algorun

ADD ./algorun_info /home/algorithm/web/algorun_info/
ADD ./src /home/algorithm/src/

# Install any algorithm dependencies here
RUN apt-get update && apt-get install -y build-essential
RUN cd /home/algorithm/src && make
```

Source code: Dockerfile of Bowtie software

Hints:

1. Dockerfile syntax requires preceding all commands with RUN keyword.
2. To ensure successful installation, always use apt-get update before installing packages and use -y option in the install command.
3. Change to /home/algorithm/src directory before running any command that operates on the source files inside src folder.

STEP 3: *manifest.json* file is required to describe the computational algorithm. Comments in the file will guide you to fill the correct values. Below is how the manifest of Bowtie looks like.

```
{
  "manifest_version": "1.2",
  "algo_name": "Bowtie 1.1.2",
  "algo_summary": "Bowtie is an ultrafast, memory-efficient alignment program for aligning short DNA sequence reads to large genomes.",
  "algo_description": "Bowtie is an ultrafast, memory-efficient short read aligner geared toward quickly aligning large sets of short DNA sequences (reads) to large genomes. Check <a href='http://bowtie-bio.sf.net/' target='_blank'>our website</a> for detailed explanation. This interface is meant to provide a quick and easy access to the computation without having to install Bowtie packages. Command line options are exposed as parameters, which you can configure from the above window.",
  "algo_website": "http://bowtie-bio.sf.net/",
  "algo_keywords": ["bowtie", "DNA", "genome", "sequencing", "alignment", "Burrows-Wheeler", "indexing"],
  "algo_authors": [
    {
      "name": "Ben Langmead",
      "email": "langmead@cs.umd.edu",
      "profile_picture": "ben.jpg",
      "personal_website": "http://www.cs.jhu.edu/~langmea/",
      "organization": "John Hopkins University",
      "org_website": "https://www.jhu.edu/"
    },
    {
      "name": "Cole Trapnell",
      "email": "coletrap@uw.edu",
      "profile_picture": "cole.png",
      "personal_website": "http://cole-trapnell-lab.github.io/team/cole-trapnell/",
      "organization": "University of Washington",
      "org_website": "http://www.uw.edu/"
    }
  ],
  "algo_exec": "ruby bowtie.rb",
  "algo_input_stream": "direct",
  "algo_output_stream": "stdout",
  "algo_parameters": {},
  "input_type": "algorun:dna-sequence",
  "output_type": "algorun:aligned-dna-sequence",
  "algo_image": "algorun/bowtie"
}
```

Source Code: manifest.json of Bowtie software (comments-skimmed)

STEP 4:

- input_example.txt file includes a sample input data for users to quickly try the algorithm. Enter ATGCATCATGCGCCAT as an example.
- **output_example.txt file includes a sample of the expected output for the same input. It makes it easier for users to expect t**

  0 - gi|110640213|ref|NC_008253.1| 148810 ATGGCGCATGATGCAT IIIIIIIIIIIIIIII 0 10:A>G,13:C>G

Notes:

- Bowtie source code comes with e_coli index packaged by default. So, use it in the algo_exec. If you included other indexes, it's ok to use them as well.
- Use direct in algo_input_stream to accept input directly from the command line. Bowtie has other options to read the input from a file. However, AlgoRun will automatically present an option to upload a file to the input area in the web interface.
- Use stdout in algo_output_stream to let AlgoRun get the result from the terminal. Bowtie has other options to write the output to a file. However, AlgoRun will automatically present an option to download the result to a file from the web interface.

STEP 5:

- **From the directory where the Dockerfile exists, build Bowtie container using:** *docker build -t bowtie .*
- You should see a success message as in the following picture.

```
 ---> 0788ef071b7e
Removing intermediate container e6d52cdf612c
Step 6 : MAINTAINER Abdelrahman Hosny <abdelrahman.hosny@hotmail.com>
 ---> Running in 5c9d49c5c70f
 ---> 6e1bfa3d638f
Removing intermediate container 5c9d49c5c70f
Successfully built 6e1bfa3d638f
abdelrahman@abdelrahman-laptop:~/uchc/algorun/examples/bowtie-1.1.2$
```

Bowtie container build success message

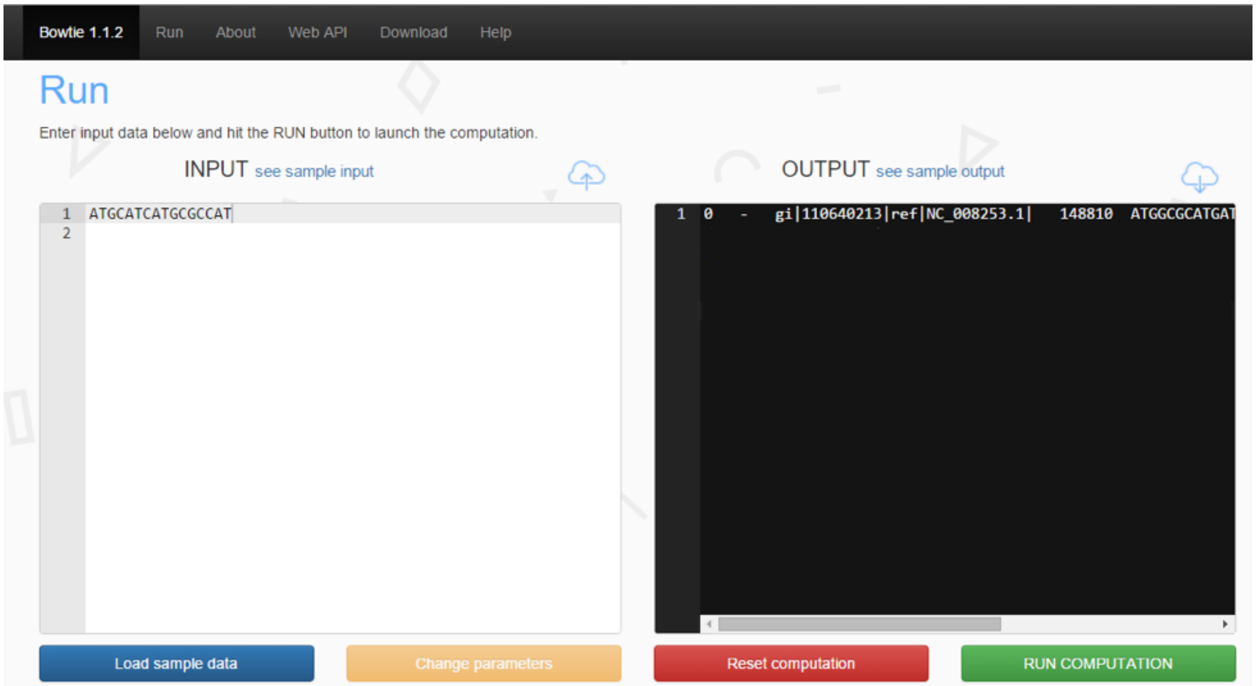
User Interface

Run Bowtie container using:

```
docker run -p 31331:8765 bowtie
```

Open the web browser and type http://localhost:31331

Hint: You can use any available port other than 31331. Yet, you must bind it to 8765 port as it is the gateway to AlgoRun.



[OPTIONAL] Expose Command Line Options as Parameters

To give flexibility to an implemented algorithm, AlgoRun allows exposing parameters that can be easily changed from the web interface. These parameters will be available as environment variables in the source code.

The power of Bowtie as a very fast DNA sequences aligner comes from the available command line options. So, you can make use of AlgoRun parameters to expose these command line options. You have two options: either to manipulate the source code of Bowtie so that it reads options from environment variables (instead of command line) or to develop a wrapper around Bowtie main executable that will internally translate environment variables to command line options. To do so, follow the below steps:

```
"algo_parameters": {
  "Skip": "0",
  "Only-Align": "all",
  "Trim-Left": "0",
  "Trim-Right": "0",
  "Phred-Quality": "33",
  "Solexa": "off",
  "Align-v": "0",
  "Align-n": "2",
  "Align-e": "70",
  "Align-l": "28",
  "Align-I": "0",
  "Align-X": "250",
  "Report-k": "1",
  "Report-all": "off",
  "Report-m": "no-limit",
  "Report-best": "off",
  "Report-strata": "off",
  "suppress": "0"
},
```

1. Specify parameters and their default values in the manifest file. The adjacent picture shows some parameters.
2. Read the input data. The input data is passed as the first command line argument.
3. Read the environment variables (of the same names you specified in the manifest) and form the options string. Call the executable file and to print the output to the standard output.

Modify the Dockerfile to install ruby dependency:

```
RUN apt-get update && apt-get install -y ruby build-essential
```

Modify algo_exec value in the manifest file to:

```
"algo_exec": "ruby bowtie.rb",
```

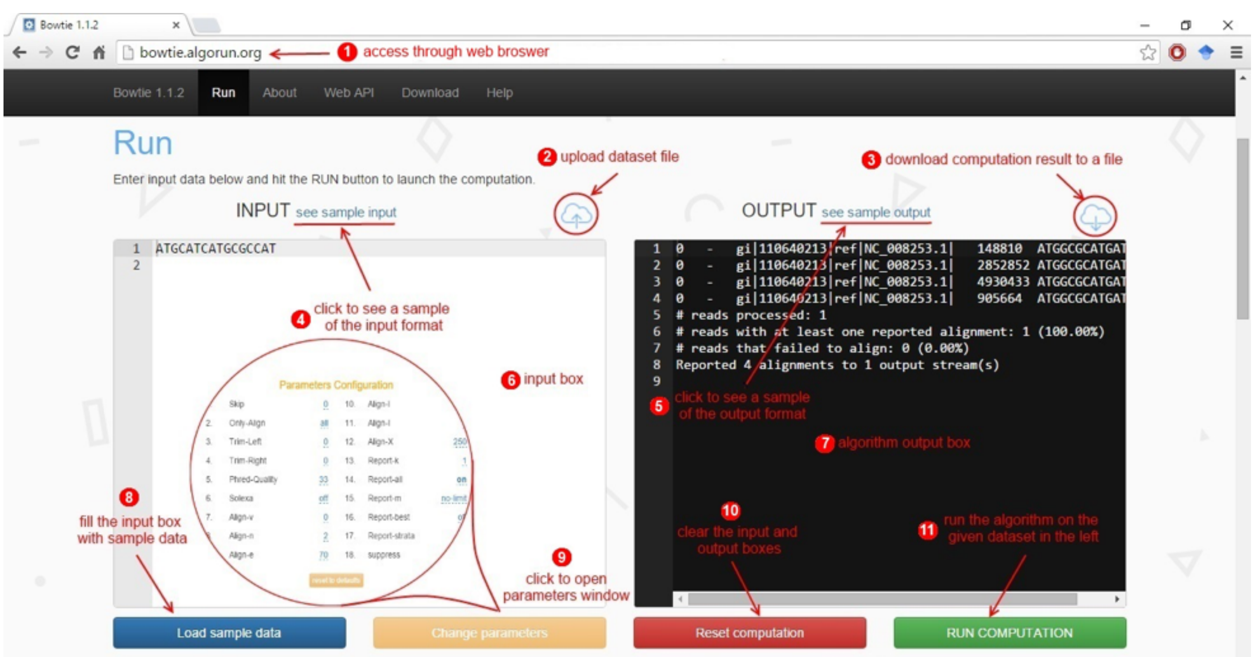
Rebuild Bowtie container using:

```
`docker build -t bowtie .`
```

At this point, options available from Bowtie can be changed by clicking on “Change Parameters” button from the web interface. Visit http://bowtie.algorun.org for the final version of Bowtie running inside AlgoRun standard container. Find the complete example on AlgoRun GitHub repository (https://github.com/algorun/algorun/tree/master/examples/bowtie-1.1.2).

```ruby
require 'open3'

# read input data that is passed directly
input_data=ARGV[0].strip

# form the options string by reading environment variables
options = ""

options += " -s " + ENV["Skip"].strip
options += " -u " + ENV["Only-Align"].strip unless ENV["Only-Align"] == "all"
options += " -5 " + ENV["Trim-Left"].strip
options += " -3 " + ENV["Trim-Right"].strip
options += " --phred64-quals" if ENV["Phred-Quality"] == "64"
options += " --solexa-quals" if ENV["Solexa"] == "on"
options += " -n " + ENV["Align-n"].strip
options += " -v " + ENV["Align-v"].strip
options += " -n " + ENV["Align-n"].strip
options += " -e " + ENV["Align-e"].strip
options += " -l " + ENV["Align-l"].strip
options += " -I " + ENV["Align-I"].strip
options += " -X " + ENV["Align-X"].strip
options += " -k " + ENV["Report-k"].strip
options += " --all" if ENV["Report-all"] == "on"
options += " -m " + ENV["Report-m"].strip unless ENV["Report-m"] == "no-limit"
options += " --best" if ENV["Report-best"] == "on"
options += " --strata" if ENV["Report-strata"] == "on"
options += " --suppress " + ENV["suppress"].delete(' ') unless ENV["suppress"] == "0"
options.strip!

# run the algorithm with the options injected
command = "./bowtie " + options + " e_coli -c " + input_data
stdin, stdout, stderr, wait_thr = Open3.popen3(command)

# print the output to the standard output stream
puts stdout.read
puts stderr.read
```

Source Code: bowtie.rb wrapper code



Bowtie web interface

Input Examples (Command Line Options vs. Parameters)

1. Example Link: http://bowtie-bio.sourceforge.net/manual.shtml#example-1–a

   Command line: ./bowtie -a -v 2 e_coli –suppress 1,5,6,7 –c ATGCATCATGCGCCAT

   With AlgoRun: Change Report-all to on, Align-v to 2 and suppress to 1,5,6,7

2. Example Link: http://bowtie-bio.sourceforge.net/manual.shtml#example-2–k-3

   Command line: ./bowtie -k 3 -v 2 e_coli –suppress 1,5,6,7 –c ATGCATCATGCGCCAT

   With AlgoRun: Change Report-k to 3, Align-v to 2 and suppress to 1,5,6,7

3. Example Link: http://bowtie-bio.sourceforge.net/manual.shtml#example-3–k-6

   Command line: ./bowtie -k 6 -v 2 e_coli –suppress 1,5,6,7 -c ATGCATCATGCGCCAT

   With AlgoRun: Change Report-k to 6, Align-v to 2 and suppress to 1,5,6,7

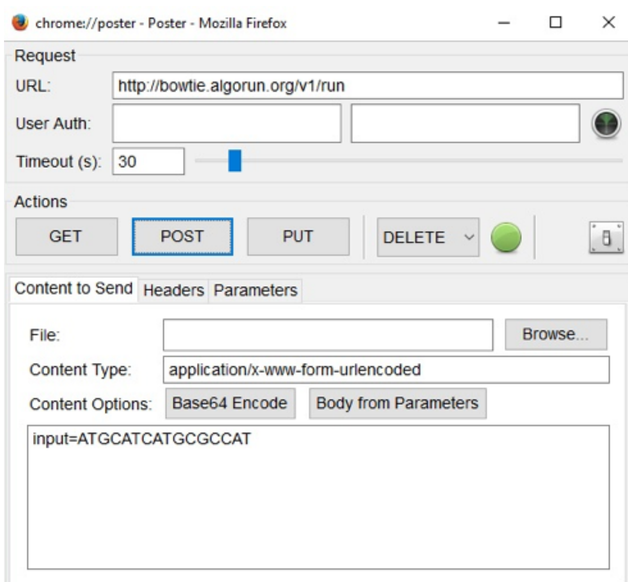
4. Example Link: http://bowtie-bio.sourceforge.net/manual.shtml#example-9–a–m-3—best—strata

   Command line: ./bowtie -a -m 3 –best –strata -v 2 e_coli –suppress 1,5,6,7 -c ATGCATCATGCGC-CAT

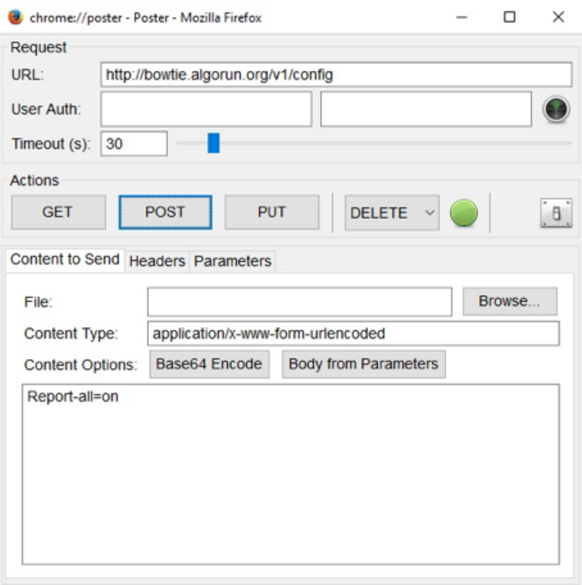
   With AlgoRun: Change Report-all to on, Report-m to 3, Report-best to on, Report-strata to on, Align-v to 2 and suppress to 1,5,6,7.

Running Bowtie through AlgoRun's Web API

In addition to the web user interface available at http://bowtie.algorun.org, you can run Bowtie using the web API. Using the web API is useful to perform the computation from different client applications. As an example of running Bowtie through the web API, see the Firefox Poster plugin examples below. Refer to supplementary file 1 for a detailed illustration on web APIs.



Run Bowtie Computation: (1) Type the URL of the endpoint http://bowtie.algorun.org/v1/run (2) Select Body from Parameters. (3) Type input=ATGCATCATGCGCCAT. (4) Click Post to initiate the request



Configure Bowtie Parameters: (1) Type the URL of the endpoint http://bowtie.algorun.org/v1/config (2) Select Body from Parameters. (3) Type Report-all=on. (4) Click Post to initiate the request.

2.2.2 Packaging REACT Algorithm with AlgoRun

REACT[2] (Vera-Licona, 2014), is a software tool to reverse engineer gene regulatory networks from time series data. The source code is written in C++ and is available on GitHub at: https://github.com/veralicona/REACT/tree/master/src In addition, the source code includes ruby files as a helper to run the algorithm.

STEP 1: Add all REACT source files inside the src folder.

STEP 2: Add the instructions to install the C++ and ruby dependencies as well as the instructions to build REACT source code to the Dockerfile. Use the helper *ruby file ruby /home/algorithm/src/run.rb make*

```
FROM algorun/algorun
ADD ./algorun_info /home/algorithm/web/algorun_info/
ADD ./src /home/algorithm/src/

# Install any algorithm dependencies here
RUN apt-get update && \
apt-get install -y build-essential ruby
RUN ruby /home/algorithm/src/run.rb make
```

Source code: Dockerfile of REACT algorithm

STEP 3: manifest.json describes REACT algorithm. Below is how the manifest of REACT looks like.

```
{
  "manifest_version": "1.2",
  "algo_name": "REACT",
  "algo_summary": "Evolutionary Algorithm for Discrete Dynamical System Optimization",
  "algo_description": "The inference of gene regulatory networks (GRNs) from system-level experimental observations is at the heart of systems
biology due to its centrality in gaining insight into the complex regulatory mechanisms in cellular systems. This includes the inference of both
the network topology and dynamic mathematical models.",
  "algo_website": "http://compsysmed.org/Software/EARevEng/REACT.html",
  "algo_keywords": ["reverse engineering", "cell biology"],
  "algo_authors": [
    {
      "name": "Paola Vera-Licona",
      "email": "veralicona@uchc.edu",
      "profile_picture": "",
      "personal_website": "http://compsysmed.org",
      "organization": "Center for Quantitative Medicine, UConn Health",
      "org_website": "http://cqm.uchc.edu/"
    },
    {
      "name": "John J. McGee",
      "email": "",
      "profile_picture": "",
      "personal_website": "",
      "organization": "Wolfram Alpha",
      "org_website": "http://www.wolframalpha.com/"
    }
  ],
  "algo_exec": "ruby React.rb",
  "algo_input_stream": "file",
  "algo_output_stream": "output.txt",
  "algo_parameters": {},
  "input_type": "[superAdam:TimeSeriesSet,superAdam:PolynomialDynamicalSystemSet,(superAdam:DirectedGraph)]",
  "output_type": "superAdam:PolynomialDynamicalSystemSet",
  "algo_image": "algorun/react"
}
```

Source Code: manifest.json of REACT algorithm (comments-skimmed)

STEP 4:

- input_example.txt file includes a sample input data for users to quickly try the algorithm. Copy and paste an example from: http://react.algorun.org/algorun_info/input_example.txt
- output_example.txt file includes a sample of the expected output for the same input. The above input produces an output of the format: http://react.algorun.org/algorun_info/output_example.txt

STEP 5:

- From the directory where the Dockerfile exists, build REACT container using:

[EXTRA STEP] Expose REACT Parameters:

REACT algorithm uses default values for different parameters. To expose these parameters to the user, include them in the manifest file in the "algo_parameter" key as in the below picture.

```
"algo_parameters": {
  "HammingPolyWeight": "0.5",
  "ComplexityWeight": "0.2",
  "RevEngWeight": "0",
  "BioProbWeight": "0",
  "HammingModelWeight": "0.35",
  "PolyScoreWeight": "0.65",
  "GenePoolSize": "100",
  "NumCandidates": "55",
  "NumParentsToPreserve": "5",
  "MaxGenerations": "100",
  "StableGenerationLimit": "50",
  "MutateProbability": "0.5"
},
```

REACT parameters in the manifest file

Parameters can be changed by clicking on "Change Parameters" button from the web interface. Visit http://react.algorun.org for the final version of REACT running inside AlgoRun standard container.

Find the complete example on AlgoRun GitHub repository (https://github.com/algorun/algorun/tree/master/examples/REACT).

2.2.3 Packaging KS Algorithm with AlgoRun

KS[3] Kavvadias-Stavropoulos algorithm (Kavvadias, 2005) generates all minimal hitting sets (traversals) of a hypergraph. The source code is written in Pascal and is available on Murakami and Uno's repository.

As the source code is using a dialect of Pascal that is not compatible with the modern compiler, download a helper executable that has been written to come over that problem: https://github.com/algorun/algorun/tree/master/examples/ks For your convenience, the repository above includes KS algorithm source code as well.

STEP 1: Add KS thg.pas source file with the helper mhs file inside the src folder. The helper

STEP 2: Add the instructions to install the C++ dependencies and Pascal compiler to the Dockerfile. As the helper is written in python, add the instructions to install the python dependencies as well. After that, navigate to the src directory and compile the source code file pc thg.pas

Adding the instructions to install *python-pip* dependency helps in installing other python packages as simplejson in a much easier way.

```
FROM algorun/algorun:latest
ADD ./src /home/algorithm/src/
ADD ./algorun_info /home/algorithm/web/algorun_info/

# Install any algorithm dependecies here
RUN apt-get update && \
    apt-get install -y --no-install-recommends \
    build-essential \
    fp-compiler \
    python-pip \
    && rm -rf /var/lib/apt/lists/
RUN pip install \
    simplejson
WORKDIR /home/algorithm/src/alg
RUN pc thg.pas
```

Source Code: Dockerfile of KS algorithm

STEP 3: manifest.json describes KS algorithm. Below is how the manifest of KS looks like.

```
{
    "manifest_version": "1.2",
    "algo_name": "KS",
    "algo_summary": "KS algorithm for minimal hitting set computations",
    "algo_description": "Introduced in <a href=\"//doi.org/10.7155/jgaa.00107\">An efficient algorithm for the transversal hypergraph generation</a>
by Kavvadias and Stavropoulos. Container sources available at the CompSysMed group <a href=\"//github.com/VeraLiconaResearchGroup/MHSGenerationAlgorit
hms/tree/master/containers/ks\">Github page</a>.",
    "algo_website": "http://lca.ceid.upatras.gr/~estavrop/transversal/",
    "algo_keywords": ["minimum hitting set", "hypergraph transversal", "MHS"],
    "algo_authors": [
        {
            "name": "Elias C. Stavropoulos",
            "email": "estavrop@ceid.upatras.gr",
            "organization": "Department of Computer Engineering and Informatics, University of Patras",
            "org_website": "www.ceid.upatras.gr"
        },
        {
            "name": "Dimitris J. Kavvadias",
            "email": "kavadias@ceid.upatras.gr",
            "organization": "Department of Computer Engineering and Informatics, University of Patras",
            "org_website": "www.ceid.upatras.gr"
        }
    ],
    "algo_exec": "./mhs",
    "algo_input_stream": "file",
    "algo_output_stream": "out.dat",
    "input_type": "algorun:mhs-in",
    "output_type": "algorun:mhs-out",
    "algo_image": "compsysmed/ks"
}
```

Source Code: manifest.json of KS algorithm (comments-skimmed)

STEP 4:

- input_example.txt: Copy and paste the following sample input:

```
{
    "sets": [
            [1, 2, 5],
            [3, 2, 4],
            [1, 3]
        ]
}
```

- output_example.txt: Copy and paste the following sample output:

```
{"transversals": [[2], [3, 4], [4, 5]], "timeTaken": 0.002045721, "sets": [[1, 2, 5],␣
↪[3, 2, 4], [1, 3]]}
```

STEP 5:

- From the directory where the Dockerfile exists, build KS container using:

```
docker build -t ks .
```

References

2.3 Input/Output Types

2.3.1 algorun:DNASequence

Usage: A Short DNA Sequence

Example:

```
ATGCATCATGCGCCAT
```

2.3.2 algorun:AlignedDNASequence

Usage: A short DNA sequence algined to large genome

Example:

```
0   -       gi|110640213|ref|NC_008253.1|   148810  ATGGCGCATGATGCAT       ␣
↪IIIIIIIIIIIIIII        0       10:A>G,13:C>G
```

2.3.3 superadam:PolynomialDynamicalSystemSet

Usage: A Set of polynomial dynamical systems

Example:

```
{
"description": "Polynomial model"",
"fieldCardinality": 2,
"name": "priorModel",
"type": "PolynomialDynamicalSystemSet",
"updateRules": [
    {
      "functions": [
          {
            "inputVariables": ["x1","x2"],
            "polynomialFunction": "x1*x2"
          }
        ],
      "target": "x1"
    },
    {
      "functions": [
          {
            "inputVariables": ["x1"],
            "polynomialFunction": "x1+1"
          },
          {
            "inputVariables": ["x1","x2"],
            "polynomialFunction": "x1*x2"
          }
        ],
      "target": "x2"
    },
    {
      "functions": [
          {
            "inputVariables": ["x1","x2","x3"],
            "polynomialFunction": "x3^2+x1"
          }
        ],
      "target": "x3"
    },
    {
      "functions": [
          {
            "inputVariables": ["x1","x3"],
            "polynomialFunction": "x3^2+x1+x2"
```

(continues on next page)

(continued from previous page)

```
          }
        ],
      "target": "x4"
    }
  ]
  }
```

2.3.4 superadam:DiscreteDynamicalSystemSet

Usage: a set of discrete dynamical systems

Example:

```
{
    "type": "DiscreteDynamicalSystemSet",
    "description": "a description",
    "simulationName": "a name",
    "updateRules": [
        {
            "target": "CAP",
            "functions": [
                {
                    "inputVariables": ["CAP"],
                    "transitionTable": [
                            [[0],0],
                            [[1],1]
                        ]
                }
            ]
        },
        {
            "target": "mRNA",
            "functions": [
                {
                    "inputVariables": ["CAP","LacI","mRNA"],
                    "transitionTable": [
                            [[0,0,0],0],
                            [[0,0,1],1],
                            [[0,1,0],0],
                            [[0,1,1],0],
                            [[0,2,0],0],
                            [[0,2,1],0],
                            [[1,0,0],1],
                            [[1,0,1],0],
                            [[1,1,0],0],
                            [[1,1,1],0],
                            [[1,2,0],0],
                            [[1,2,1],0]
                      ]
                }
             ]
        },
        {
            "target": "LacY",
            "functions": [
```

(continues on next page)

(continued from previous page)

```
                {
                    "inputVariables": ["mRNA","LacY"],
                    "transitionTable": [
                            [[0,0],0],
                            [[0,1],1],
                            [[1,0],1],
                            [[1,1],1]
                        ]
                 }
            ]
        },
        {
            "target": "LacZ",
            "functions": [
                {
                    "inputVariables": ["mRNA","LacZ"],
                    "transitionTable": [
                            [[0,0],0],
                            [[0,1],1],
                            [[1,0],1],
                            [[1,1],1]
                         ]
                }
            ]
        },
        {
            "target": "LacI",
            "functions": [
                {
                    "inputVariables": ["LacI"],
                    "transitionTable": [
                            [[0],0],
                            [[1],1],
                            [[2],2]
                        ]
                }
            ]
        }
    ],
    "variables": [
        {
            "id": "CAP",
            "states": [0,1],
            "speed": 1
        },
        {
            "id": "mRNA",
            "states": [0,1],
            "speed": 1
        },
        {
            "id": "LacY",
            "states": [0,1],
            "speed": 1
        },
        {
            "id": "LacZ",
```

(continues on next page)

(continued from previous page)

```
            "states": [0,1],
            "speed": 1
        },
        {
            "id": "LacI",
            "states": [0,1,2],
            "speed": 1
        }
    ]
}
```

2.3.5 superadam:BooleanDynamicalSystemSet

Usage: a set of boolean dynamical systems

Example:

```
{
    "type": "BooleanDynamicalSystemSet",
    "description": "Sample Boolean Network",
    "parameters": [
        {
        "id": "k1",
        "states": [0,1]
        },
        {
        "id": "k2",
        "states": [0,1]
        }
      ],
    "updateRules": [
        {
            "target": "x1",
            "functions": [
                {
                    "inputVariables": ["k1","x3"],
                    "booleanFunction": "k1 & x3"
                }
            ]
        },
        {
            "target": "x2",
            "functions": [
                {
                    "inputVariables": ["x1" , "k2"],
                    "booleanFunction": "x1 | k2"
                }
            ]
        },
        {
            "target": "x3",
            "functions": [
                {
                    "inputVariables": ["x4","x2"],
                    "booleanFunction": "x2 & !x4"
```

(continues on next page)

(continued from previous page)

```
                }
            ]
        },
        {
            "target": "x4",
            "functions": [
                {
                    "inputVariables": ["x2","k2"],
                    "booleanFunction": "x2 & k2"
                }
            ]
        }
    ],

    "variables": [
        {
            "id": "x1",
            "states": [0,1]
        },
        {
            "id": "x2",
            "states": [0,1]
        },
        {
            "id": "x3",
            "states": [0,1]
        },
        {
            "id": "x4",
            "states": [0,1]
        }
    ]
}
```

2.3.6 superadam:TimeSeriesSet

Usage: A set of time series

Example:

```
{
"type": "timeSeriesSet",
"timeSeriesData": [
    {
      "index": [],
      "matrix": [
          [1,0,0,0],
          [0,1,0,1],
          [1,1,0,0]
      ],
      "name": "wildtype experiment 1"
    },
    {
      "index": [],
      "matrix": [
```

(continues on next page)

(continued from previous page)

```
          [1,1,0,0],
          [0,0,0,1],
          [1,0,0,0]
        ],
      "name": "wildtype experiment 2"
    },
    {
      "index": [1],
      "matrix": [
          [0,0,0,0],
          [0,0,0,1],
          [0,1,0,0],
          [0,0,0,1]
        ],
      "name": "knockout experiment 1"
    },
    {
      "index": [3],
      "matrix": [
          [0,1,0,0],
          [0,1,0,1],
          [0,1,0,1],
          [0,0,0,0]
        ],
      "name": "knockout experiment 2"
    },
    {
      "index": [2],
      "matrix": [
          [1,0,0,0],
          [0,0,0,1],
          [1,0,0,1],
          [0,0,0,0]
        ],
      "name": "knockout experiment 3"
    }
  ]
}
```

2.3.7 superadam:DirectedGraph

Usage: a directed graph representation

Example:

```
{
"description": "",
"fieldCardinality": 2,
"name": "priorReverseEngineeringNetwork",
"type": "directedGraph",
"edges": [
    {
      "sources": [
          { "score": 0.5, "source": "x1" },
          { "score": 1, "source": "x2" },
```

(continues on next page)

(continued from previous page)

```
            { "score": 1, "source": "x3" },
            { "score": 1, "source": "x4" }
          ],
        "target": "x1"
      },
      {
        "sources": [
            { "score": 1, "source": "x2" },
            { "score": 1, "source": "x4" }
          ],
        "target": "x2"
      },
      {
        "sources": [
            { "score": 0.5, "source": "x1" },
            { "score": 0.5, "source": "x2" }
          ],
        "target": "x3"
      },
      {
        "sources": [
            { "score": 0.33, "source": "x2" },
            { "score": 0.66, "source": "x3" }
          ],
        "target": "x4"
      }
]
}
```

2.3.8 superadam:AnnotatedGraph

Usage: an annotated graph representation

Example:

```
{
    "type": "AnnotatedGraph",
    "description": "Sample Annotated Graph",
    "node": [
       {
          "id" : "node0",
          "label": " 0 0 0 0 0"
       },
       {
          "id" : "node1",
          "label": " 0 0 0 0 1"
       },
       {
          "id" : "node2",
          "label": " 0 0 0 0 2"
       },
       {
          "id" : "node3",
          "label": " 0 0 0 1 0"
       },
```

(continues on next page)

(continued from previous page)

```
        {
           "id" : "node4",
           "label": " 0 0 0 1 1"
        },
        {
           "id" : "node5",
           "label": " 0 0 0 1 2"
        },
        {
           "id" : "node6",
           "label": " 0 0 1 0 0"
        },
        {
           "id" : "node7",
           "label": " 0 0 1 0 1"
        },
        {
           "id" : "node8",
           "label": " 0 0 1 0 2"
        },
        {
           "id" : "node9",
           "label": " 0 0 1 1 0"
        },
        {
           "id" : "node10",
           "label": " 0 0 1 1 1"
        },
        {
           "id" : "node11",
           "label": " 0 0 1 1 2"
        },
        {
           "id" : "node12",
           "label": " 0 1 0 0 0"
        },
        {
           "id" : "node13",
           "label": " 0 1 0 0 1"
        },
        {
           "id" : "node14",
           "label": " 0 1 0 0 2"
        },
        {
           "id" : "node15",
           "label": " 0 1 0 1 0"
        },
        {
           "id" : "node16",
           "label": " 0 1 0 1 1"
        },
        {
           "id" : "node17",
           "label": " 0 1 0 1 2"
        },
        {
```

(continues on next page)

(continued from previous page)

```
            "id" : "node18",
            "label": " 0 1 1 0 0"
        },
        {
            "id" : "node19",
            "label": " 0 1 1 0 1"
        },
        {
            "id" : "node20",
            "label": " 0 1 1 0 2"
        },
        {
            "id" : "node21",
            "label": " 0 1 1 1 0"
        },
        {
            "id" : "node22",
            "label": " 0 1 1 1 1"
        },
        {
            "id" : "node23",
            "label": " 0 1 1 1 2"
        },
        {
            "id" : "node24",
            "label": " 1 0 0 0 0"
        },
        {
            "id" : "node25",
            "label": " 1 0 0 0 1"
        },
        {
            "id" : "node26",
            "label": " 1 0 0 0 2"
        },
        {
            "id" : "node27",
            "label": " 1 0 0 1 0"
        },
        {
            "id" : "node28",
            "label": " 1 0 0 1 1"
        },
        {
            "id" : "node29",
            "label": " 1 0 0 1 2"
        },
        {
            "id" : "node30",
            "label": " 1 0 1 0 0"
        },
        {
            "id" : "node31",
            "label": " 1 0 1 0 1"
        },
        {
            "id" : "node32",
```

(continues on next page)

(continued from previous page)

```
            "label": " 1 0 1 0 2"
        },
        {
            "id" : "node33",
            "label": " 1 0 1 1 0"
        },
        {
            "id" : "node34",
            "label": " 1 0 1 1 1"
        },
        {
            "id" : "node35",
            "label": " 1 0 1 1 2"
        },
        {
            "id" : "node36",
            "label": " 1 1 0 0 0"
        },
        {
            "id" : "node37",
            "label": " 1 1 0 0 1"
        },
        {
            "id" : "node38",
            "label": " 1 1 0 0 2"
        },
        {
            "id" : "node39",
            "label": " 1 1 0 1 0"
        },
        {
            "id" : "node40",
            "label": " 1 1 0 1 1"
        },
        {
            "id" : "node41",
            "label": " 1 1 0 1 2"
        },
        {
            "id" : "node42",
            "label": " 1 1 1 0 0"
        },
        {
            "id" : "node43",
            "label": " 1 1 1 0 1"
        },
        {
            "id" : "node44",
            "label": " 1 1 1 0 2"
        },
        {
            "id" : "node45",
            "label": " 1 1 1 1 0"
        },
        {
            "id" : "node46",
            "label": " 1 1 1 1 1"
```

(continues on next page)

(continued from previous page)

```
        },
        {
            "id" : "node47",
            "label": " 1 1 1 1 2"
        }
    ],
    "connection": [
         ["node0", "node0"],
         ["node1", "node1"],
         ["node2","node2"],
         ["node3","node3"],
         ["node4","node4"],
         ["node5","node5"],
         ["node6","node6"],
         ["node7","node7"],
         ["node8","node8"],
         ["node9","node9"],
         ["node10","node10"],
         ["node11","node11"],
         ["node12","node21"],
         ["node13","node10"],
         ["node14","node11"],
         ["node15","node21"],
         ["node16","node10"],
         ["node17","node11"],
         ["node18","node21"],
         ["node19","node10"],
         ["node20","node11"],
         ["node21","node21"],
         ["node22","node10"],
         ["node23","node11"],
         ["node24","node36"],
         ["node25","node25"],
         ["node26","node26"],
         ["node27","node39"],
         ["node28","node28"],
         ["node29","node29"],
         ["node30","node42"],
         ["node31","node31"],
         ["node32","node32"],
         ["node33","node45"],
         ["node34","node34"],
         ["node35","node35"],
         ["node36","node33"],
         ["node37","node34"],
         ["node38","node35"],
         ["node39","node33"],
         ["node40","node34"],
         ["node41","node35"],
         ["node42","node33"],
         ["node43","node34"],
         ["node44","node35"],
         ["node45","node33"],
         ["node46","node34"],
         ["node47","node35"]
    ]
    }
```

2.3.9 superadam:SteadyStates

Usage: steady states of boolean dynamical system set

Example:

```
{
"type": "SteadyStates",
"description": "steady states of boolean dynamical system set",
"steadystates": {
            "idorder": ["x1","x2","x3","x4"],
            "value": [
                        [0,0,0,0],
                        [1,1,1,0]
                     ]
                }
    }
```

2.4 License

GNU GENERAL PUBLIC LICENSE Version 3, 29 June 2007

> Copyright (C) 2007 Free Software Foundation, Inc. <http://fsf.org/> Everyone is permitted to copy and distribute verbatim copies of this license document, but changing it is not allowed.

Preamble

> The GNU General Public License is a free, copyleft license for

software and other kinds of works.

> The licenses for most software and other practical works are designed

to take away your freedom to share and change the works. By contrast, the GNU General Public License is intended to guarantee your freedom to share and change all versions of a program–to make sure it remains free software for all its users. We, the Free Software Foundation, use the GNU General Public License for most of our software; it applies also to any other work released this way by its authors. You can apply it to your programs, too.

> When we speak of free software, we are referring to freedom, not

price. Our General Public Licenses are designed to make sure that you have the freedom to distribute copies of free software (and charge for them if you wish), that you receive source code or can get it if you want it, that you can change the software or use pieces of it in new free programs, and that you know you can do these things.

> To protect your rights, we need to prevent others from denying you

these rights or asking you to surrender the rights. Therefore, you have certain responsibilities if you distribute copies of the software, or if you modify it: responsibilities to respect the freedom of others.

> For example, if you distribute copies of such a program, whether

gratis or for a fee, you must pass on to the recipients the same freedoms that you received. You must make sure that they, too, receive or can get the source code. And you must show them these terms so they know their rights.

> Developers that use the GNU GPL protect your rights with two steps:

(1) assert copyright on the software, and (2) offer you this License giving you legal permission to copy, distribute and/or modify it.

> For the developers' and authors' protection, the GPL clearly explains

that there is no warranty for this free software. For both users' and authors' sake, the GPL requires that modified versions be marked as changed, so that their problems will not be attributed erroneously to authors of previous versions.

Some devices are designed to deny users access to install or run

modified versions of the software inside them, although the manufacturer can do so. This is fundamentally incompatible with the aim of protecting users' freedom to change the software. The systematic pattern of such abuse occurs in the area of products for individuals to use, which is precisely where it is most unacceptable. Therefore, we have designed this version of the GPL to prohibit the practice for those products. If such problems arise substantially in other domains, we stand ready to extend this provision to those domains in future versions of the GPL, as needed to protect the freedom of users.

Finally, every program is threatened constantly by software patents.

States should not allow patents to restrict development and use of software on general-purpose computers, but in those that do, we wish to avoid the special danger that patents applied to a free program could make it effectively proprietary. To prevent this, the GPL assures that patents cannot be used to render the program non-free.

The precise terms and conditions for copying, distribution and

modification follow.

TERMS AND CONDITIONS

0. Definitions.

"This License" refers to version 3 of the GNU General Public License.

"Copyright" also means copyright-like laws that apply to other kinds of

works, such as semiconductor masks.

"The Program" refers to any copyrightable work licensed under this

License. Each licensee is addressed as "you". "Licensees" and "recipients" may be individuals or organizations.

To "modify" a work means to copy from or adapt all or part of the work

in a fashion requiring copyright permission, other than the making of an exact copy. The resulting work is called a "modified version" of the earlier work or a work "based on" the earlier work.

A "covered work" means either the unmodified Program or a work based

on the Program.

To "propagate" a work means to do anything with it that, without

permission, would make you directly or secondarily liable for infringement under applicable copyright law, except executing it on a computer or modifying a private copy. Propagation includes copying, distribution (with or without modification), making available to the public, and in some countries other activities as well.

To "convey" a work means any kind of propagation that enables other

parties to make or receive copies. Mere interaction with a user through a computer network, with no transfer of a copy, is not conveying.

An interactive user interface displays "Appropriate Legal Notices"

to the extent that it includes a convenient and prominently visible feature that (1) displays an appropriate copyright notice, and (2) tells the user that there is no warranty for the work (except to the extent that warranties are provided), that licensees may convey the work under this License, and how to view a copy of this License. If the interface presents a list of user commands or options, such as a menu, a prominent item in the list meets this criterion.

1. Source Code.

The "source code" for a work means the preferred form of the work for making modifications to it. "Object code" means any non-source form of a work.

A "Standard Interface" means an interface that either is an official

standard defined by a recognized standards body, or, in the case of interfaces specified for a particular programming language, one that is widely used among developers working in that language.

The "System Libraries" of an executable work include anything, other

than the work as a whole, that (a) is included in the normal form of packaging a Major Component, but which is not part of that Major Component, and (b) serves only to enable use of the work with that Major Component, or to implement a Standard Interface for which an implementation is available to the public in source code form. A "Major Component", in this context, means a major essential component (kernel, window system, and so on) of the specific operating system (if any) on which the executable work runs, or a compiler used to produce the work, or an object code interpreter used to run it.

The "Corresponding Source" for a work in object code form means all

the source code needed to generate, install, and (for an executable work) run the object code and to modify the work, including scripts to control those activities. However, it does not include the work's System Libraries, or general-purpose tools or generally available free programs which are used unmodified in performing those activities but which are not part of the work. For example, Corresponding Source includes interface definition files associated with source files for the work, and the source code for shared libraries and dynamically linked subprograms that the work is specifically designed to require, such as by intimate data communication or control flow between those subprograms and other parts of the work.

The Corresponding Source need not include anything that users

can regenerate automatically from other parts of the Corresponding Source.

The Corresponding Source for a work in source code form is that

same work.

2. Basic Permissions.

All rights granted under this License are granted for the term of

copyright on the Program, and are irrevocable provided the stated conditions are met. This License explicitly affirms your unlimited permission to run the unmodified Program. The output from running a covered work is covered by this License only if the output, given its content, constitutes a covered work. This License acknowledges your rights of fair use or other equivalent, as provided by copyright law.

You may make, run and propagate covered works that you do not

convey, without conditions so long as your license otherwise remains in force. You may convey covered works to others for the sole purpose of having them make modifications exclusively for you, or provide you with facilities for running those works, provided that you comply with the terms of this License in conveying all material for which you do not control copyright. Those thus making or running the covered works for you must do so exclusively on your behalf, under your direction and control, on terms that prohibit them from making any copies of your copyrighted material outside their relationship with you.

Conveying under any other circumstances is permitted solely under

the conditions stated below. Sublicensing is not allowed; section 10 makes it unnecessary.

3. Protecting Users' Legal Rights From Anti-Circumvention Law.

No covered work shall be deemed part of an effective technological

measure under any applicable law fulfilling obligations under article 11 of the WIPO copyright treaty adopted on 20 December 1996, or similar laws prohibiting or restricting circumvention of such measures.

When you convey a covered work, you waive any legal power to forbid circumvention of technological measures to the extent such circumvention is effected by exercising rights under this License with respect to the covered work, and you disclaim any intention to limit operation or modification of the work as a means of enforcing, against the work's users, your or third parties' legal rights to forbid circumvention of technological measures.

4. Conveying Verbatim Copies.

   You may convey verbatim copies of the Program's source code as you

receive it, in any medium, provided that you conspicuously and appropriately publish on each copy an appropriate copyright notice; keep intact all notices stating that this License and any non-permissive terms added in accord with section 7 apply to the code; keep intact all notices of the absence of any warranty; and give all recipients a copy of this License along with the Program.

   You may charge any price or no price for each copy that you convey,

and you may offer support or warranty protection for a fee.

5. Conveying Modified Source Versions.

   You may convey a work based on the Program, or the modifications to

produce it from the Program, in the form of source code under the terms of section 4, provided that you also meet all of these conditions:

> a) The work must carry prominent notices stating that you modified it, and giving a relevant date.
>
> b) The work must carry prominent notices stating that it is released under this License and any conditions added under section 7. This requirement modifies the requirement in section 4 to "keep intact all notices".
>
> c) You must license the entire work, as a whole, under this License to anyone who comes into possession of a copy. This License will therefore apply, along with any applicable section 7 additional terms, to the whole of the work, and all its parts, regardless of how they are packaged. This License gives no permission to license the work in any other way, but it does not invalidate such permission if you have separately received it.
>
> d) If the work has interactive user interfaces, each must display Appropriate Legal Notices; however, if the Program has interactive interfaces that do not display Appropriate Legal Notices, your work need not make them do so.

A compilation of a covered work with other separate and independent

works, which are not by their nature extensions of the covered work, and which are not combined with it such as to form a larger program, in or on a volume of a storage or distribution medium, is called an "aggregate" if the compilation and its resulting copyright are not used to limit the access or legal rights of the compilation's users beyond what the individual works permit. Inclusion of a covered work in an aggregate does not cause this License to apply to the other parts of the aggregate.

6. Conveying Non-Source Forms.

   You may convey a covered work in object code form under the terms

of sections 4 and 5, provided that you also convey the machine-readable Corresponding Source under the terms of this License, in one of these ways:

> a) Convey the object code in, or embodied in, a physical product (including a physical distribution medium), accompanied by the Corresponding Source fixed on a durable physical medium customarily used for software interchange.
>
> b) Convey the object code in, or embodied in, a physical product (including a physical distribution medium), accompanied by a written offer, valid for at least three years and valid for as

long as you offer spare parts or customer support for that product model, to give anyone who possesses the object code either (1) a copy of the Corresponding Source for all the software in the product that is covered by this License, on a durable physical medium customarily used for software interchange, for a price no more than your reasonable cost of physically performing this conveying of source, or (2) access to copy the Corresponding Source from a network server at no charge.

c) Convey individual copies of the object code with a copy of the written offer to provide the Corresponding Source. This alternative is allowed only occasionally and noncommercially, and only if you received the object code with such an offer, in accord with subsection 6b.

d) Convey the object code by offering access from a designated place (gratis or for a charge), and offer equivalent access to the Corresponding Source in the same way through the same place at no further charge. You need not require recipients to copy the Corresponding Source along with the object code. If the place to copy the object code is a network server, the Corresponding Source may be on a different server (operated by you or a third party) that supports equivalent copying facilities, provided you maintain clear directions next to the object code saying where to find the Corresponding Source. Regardless of what server hosts the Corresponding Source, you remain obligated to ensure that it is available for as long as needed to satisfy these requirements.

e) Convey the object code using peer-to-peer transmission, provided you inform other peers where the object code and Corresponding Source of the work are being offered to the general public at no charge under subsection 6d.

A separable portion of the object code, whose source code is excluded

from the Corresponding Source as a System Library, need not be included in conveying the object code work.

A “User Product” is either (1) a “consumer product”, which means any

tangible personal property which is normally used for personal, family, or household purposes, or (2) anything designed or sold for incorporation into a dwelling. In determining whether a product is a consumer product, doubtful cases shall be resolved in favor of coverage. For a particular product received by a particular user, “normally used” refers to a typical or common use of that class of product, regardless of the status of the particular user or of the way in which the particular user actually uses, or expects or is expected to use, the product. A product is a consumer product regardless of whether the product has substantial commercial, industrial or non-consumer uses, unless such uses represent the only significant mode of use of the product.

“Installation Information” for a User Product means any methods,

procedures, authorization keys, or other information required to install and execute modified versions of a covered work in that User Product from a modified version of its Corresponding Source. The information must suffice to ensure that the continued functioning of the modified object code is in no case prevented or interfered with solely because modification has been made.

If you convey an object code work under this section in, or with, or

specifically for use in, a User Product, and the conveying occurs as part of a transaction in which the right of possession and use of the User Product is transferred to the recipient in perpetuity or for a fixed term (regardless of how the transaction is characterized), the Corresponding Source conveyed under this section must be accompanied by the Installation Information. But this requirement does not apply if neither you nor any third party retains the ability to install modified object code on the User Product (for example, the work has been installed in ROM).

The requirement to provide Installation Information does not include a

requirement to continue to provide support service, warranty, or updates for a work that has been modified or installed by the recipient, or for the User Product in which it has been modified or installed. Access to a network may be denied when the modification itself materially and adversely affects the operation of the network or violates the rules and protocols for communication across the network.

Corresponding Source conveyed, and Installation Information provided,

in accord with this section must be in a format that is publicly documented (and with an implementation available to the public in source code form), and must require no special password or key for unpacking, reading or copying.

7. Additional Terms.

“Additional permissions” are terms that supplement the terms of this

License by making exceptions from one or more of its conditions. Additional permissions that are applicable to the entire Program shall be treated as though they were included in this License, to the extent that they are valid under applicable law. If additional permissions apply only to part of the Program, that part may be used separately under those permissions, but the entire Program remains governed by this License without regard to the additional permissions.

When you convey a copy of a covered work, you may at your option

remove any additional permissions from that copy, or from any part of it. (Additional permissions may be written to require their own removal in certain cases when you modify the work.) You may place additional permissions on material, added by you to a covered work, for which you have or can give appropriate copyright permission.

Notwithstanding any other provision of this License, for material you

add to a covered work, you may (if authorized by the copyright holders of that material) supplement the terms of this License with terms:

> a) Disclaiming warranty or limiting liability differently from the terms of sections 15 and 16 of this License; or
>
> b) Requiring preservation of specified reasonable legal notices or author attributions in that material or in the Appropriate Legal Notices displayed by works containing it; or
>
> c) Prohibiting misrepresentation of the origin of that material, or requiring that modified versions of such material be marked in reasonable ways as different from the original version; or
>
> d) Limiting the use for publicity purposes of names of licensors or authors of the material; or
>
> e) Declining to grant rights under trademark law for use of some trade names, trademarks, or service marks; or
>
> f) Requiring indemnification of licensors and authors of that material by anyone who conveys the material (or modified versions of it) with contractual assumptions of liability to the recipient, for any liability that these contractual assumptions directly impose on those licensors and authors.

All other non-permissive additional terms are considered “further

restrictions” within the meaning of section 10. If the Program as you received it, or any part of it, contains a notice stating that it is governed by this License along with a term that is a further restriction, you may remove that term. If a license document contains a further restriction but permits relicensing or conveying under this License, you may add to a covered work material governed by the terms of that license document, provided that the further restriction does not survive such relicensing or conveying.

If you add terms to a covered work in accord with this section, you

must place, in the relevant source files, a statement of the additional terms that apply to those files, or a notice indicating where to find the applicable terms.

Additional terms, permissive or non-permissive, may be stated in the

form of a separately written license, or stated as exceptions; the above requirements apply either way.

8. Termination.

You may not propagate or modify a covered work except as expressly

provided under this License. Any attempt otherwise to propagate or modify it is void, and will automatically terminate your rights under this License (including any patent licenses granted under the third paragraph of section 11).

However, if you cease all violation of this License, then your

license from a particular copyright holder is reinstated (a) provisionally, unless and until the copyright holder explicitly and finally terminates your license, and (b) permanently, if the copyright holder fails to notify you of the violation by some reasonable means prior to 60 days after the cessation.

Moreover, your license from a particular copyright holder is

reinstated permanently if the copyright holder notifies you of the violation by some reasonable means, this is the first time you have received notice of violation of this License (for any work) from that copyright holder, and you cure the violation prior to 30 days after your receipt of the notice.

Termination of your rights under this section does not terminate the

licenses of parties who have received copies or rights from you under this License. If your rights have been terminated and not permanently reinstated, you do not qualify to receive new licenses for the same material under section 10.

9. Acceptance Not Required for Having Copies.

You are not required to accept this License in order to receive or

run a copy of the Program. Ancillary propagation of a covered work occurring solely as a consequence of using peer-to-peer transmission to receive a copy likewise does not require acceptance. However, nothing other than this License grants you permission to propagate or modify any covered work. These actions infringe copyright if you do not accept this License. Therefore, by modifying or propagating a covered work, you indicate your acceptance of this License to do so.

10. Automatic Licensing of Downstream Recipients.

Each time you convey a covered work, the recipient automatically

receives a license from the original licensors, to run, modify and propagate that work, subject to this License. You are not responsible for enforcing compliance by third parties with this License.

An "entity transaction" is a transaction transferring control of an

organization, or substantially all assets of one, or subdividing an organization, or merging organizations. If propagation of a covered work results from an entity transaction, each party to that transaction who receives a copy of the work also receives whatever licenses to the work the party's predecessor in interest had or could give under the previous paragraph, plus a right to possession of the Corresponding Source of the work from the predecessor in interest, if the predecessor has it or can get it with reasonable efforts.

You may not impose any further restrictions on the exercise of the

rights granted or affirmed under this License. For example, you may not impose a license fee, royalty, or other charge for exercise of rights granted under this License, and you may not initiate litigation (including a cross-claim or counterclaim in a lawsuit) alleging that any patent claim is infringed by making, using, selling, offering for sale, or importing the Program or any portion of it.

11. Patents.

A "contributor" is a copyright holder who authorizes use under this

License of the Program or a work on which the Program is based. The work thus licensed is called the contributor's "contributor version".

A contributor's "essential patent claims" are all patent claims

owned or controlled by the contributor, whether already acquired or hereafter acquired, that would be infringed by some manner, permitted by this License, of making, using, or selling its contributor version, but do not include claims that would be infringed only as a consequence of further modification of the contributor version. For purposes of this definition, "control" includes the right to grant patent sublicenses in a manner consistent with the requirements of this License.

Each contributor grants you a non-exclusive, worldwide, royalty-free

patent license under the contributor's essential patent claims, to make, use, sell, offer for sale, import and otherwise run, modify and propagate the contents of its contributor version.

In the following three paragraphs, a "patent license" is any express

agreement or commitment, however denominated, not to enforce a patent (such as an express permission to practice a patent or covenant not to sue for patent infringement). To "grant" such a patent license to a party means to make such an agreement or commitment not to enforce a patent against the party.

If you convey a covered work, knowingly relying on a patent license,

and the Corresponding Source of the work is not available for anyone to copy, free of charge and under the terms of this License, through a publicly available network server or other readily accessible means, then you must either (1) cause the Corresponding Source to be so available, or (2) arrange to deprive yourself of the benefit of the patent license for this particular work, or (3) arrange, in a manner consistent with the requirements of this License, to extend the patent license to downstream recipients. "Knowingly relying" means you have actual knowledge that, but for the patent license, your conveying the covered work in a country, or your recipient's use of the covered work in a country, would infringe one or more identifiable patents in that country that you have reason to believe are valid.

If, pursuant to or in connection with a single transaction or

arrangement, you convey, or propagate by procuring conveyance of, a covered work, and grant a patent license to some of the parties receiving the covered work authorizing them to use, propagate, modify or convey a specific copy of the covered work, then the patent license you grant is automatically extended to all recipients of the covered work and works based on it.

A patent license is "discriminatory" if it does not include within

the scope of its coverage, prohibits the exercise of, or is conditioned on the non-exercise of one or more of the rights that are specifically granted under this License. You may not convey a covered work if you are a party to an arrangement with a third party that is in the business of distributing software, under which you make payment to the third party based on the extent of your activity of conveying the work, and under which the third party grants, to any of the parties who would receive the covered work from you, a discriminatory patent license (a) in connection with copies of the covered work conveyed by you (or copies made from those copies), or (b) primarily for and in connection with specific products or compilations that contain the covered work, unless you entered into that arrangement, or that patent license was granted, prior to 28 March 2007.

Nothing in this License shall be construed as excluding or limiting

any implied license or other defenses to infringement that may otherwise be available to you under applicable patent law.

12. No Surrender of Others' Freedom.

    If conditions are imposed on you (whether by court order, agreement or

otherwise) that contradict the conditions of this License, they do not excuse you from the conditions of this License. If you cannot convey a covered work so as to satisfy simultaneously your obligations under this License and any other pertinent obligations, then as a consequence you may not convey it at all. For example, if you agree to terms that obligate you to collect a royalty for further conveying from those to whom you convey the Program, the only way you could satisfy both those terms and this License would be to refrain entirely from conveying the Program.

13. Use with the GNU Affero General Public License.

    Notwithstanding any other provision of this License, you have permission to link or combine any covered work with a work licensed under version 3 of the GNU Affero General Public License into a single combined work, and to convey the resulting work. The terms of this License will continue to apply to the part which is the covered work, but the special requirements of the GNU Affero General Public License, section 13, concerning interaction through a network will apply to the combination as such.

14. Revised Versions of this License.

    The Free Software Foundation may publish revised and/or new versions of

the GNU General Public License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns.

    Each version is given a distinguishing version number. If the

Program specifies that a certain numbered version of the GNU General Public License "or any later version" applies to it, you have the option of following the terms and conditions either of that numbered version or of any later version published by the Free Software Foundation. If the Program does not specify a version number of the GNU General Public License, you may choose any version ever published by the Free Software Foundation.

    If the Program specifies that a proxy can decide which future

versions of the GNU General Public License can be used, that proxy's public statement of acceptance of a version permanently authorizes you to choose that version for the Program.

    Later license versions may give you additional or different

permissions. However, no additional obligations are imposed on any author or copyright holder as a result of your choosing to follow a later version.

15. Disclaimer of Warranty.

THERE IS NO WARRANTY FOR THE PROGRAM, TO THE EXTENT PERMITTED BY APPLICABLE LAW. EXCEPT WHEN OTHERWISE STATED IN WRITING THE COPYRIGHT HOLDERS AND/OR OTHER PARTIES PROVIDE THE PROGRAM "AS IS" 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. THE ENTIRE RISK AS TO THE QUALITY AND PERFORMANCE OF THE PROGRAM IS WITH YOU. SHOULD THE PROGRAM PROVE DEFECTIVE, YOU ASSUME THE COST OF ALL NECESSARY SERVICING, REPAIR OR CORRECTION.

16. Limitation of Liability.

    IN NO EVENT UNLESS REQUIRED BY APPLICABLE LAW OR AGREED TO IN WRITING

WILL ANY COPYRIGHT HOLDER, OR ANY OTHER PARTY WHO MODIFIES AND/OR CONVEYS THE PROGRAM AS PERMITTED ABOVE, BE LIABLE TO YOU FOR DAMAGES, INCLUDING ANY GENERAL, SPECIAL, INCIDENTAL OR CONSEQUENTIAL DAMAGES ARISING OUT OF THE USE OR INABILITY TO USE THE PROGRAM (INCLUDING BUT NOT LIMITED TO LOSS OF DATA OR DATA BEING RENDERED INACCURATE OR LOSSES SUSTAINED BY YOU OR THIRD PARTIES OR A FAILURE OF THE PROGRAM TO OPERATE WITH ANY OTHER PROGRAMS), EVEN IF SUCH HOLDER OR OTHER PARTY HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

17. Interpretation of Sections 15 and 16.

    If the disclaimer of warranty and limitation of liability provided

above cannot be given local legal effect according to their terms, reviewing courts shall apply local law that most closely approximates an absolute waiver of all civil liability in connection with the Program, unless a warranty or assumption of liability accompanies a copy of the Program in return for a fee.

END OF TERMS AND CONDITIONS

How to Apply These Terms to Your New Programs

    If you develop a new program, and you want it to be of the greatest possible use to the public, the best way to achieve this is to make it free software which everyone can redistribute and change under these terms.

To do so, attach the following notices to the program. It is safest

to attach them to the start of each source file to most effectively state the exclusion of warranty; and each file should have at least the "copyright" line and a pointer to where the full notice is found.

> Algorun turns command-line algorithms into ready-to-use web enabled containers - Copyright (C) 2015 Thibauld Favre <tfavre@gmail.com>
>
> This program is free software: you can redistribute it and/or modify it under the terms of the GNU General Public License as published by the Free Software Foundation, either version 3 of the License, or (at your option) any later version.
>
> This program is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU General Public License for more details.
>
> You should have received a copy of the GNU General Public License along with this program. If not, see <http://www.gnu.org/licenses/>.

Also add information on how to contact you by electronic and paper mail.

If the program does terminal interaction, make it output a short

notice like this when it starts in an interactive mode:

> Algorun Copyright (C) 2015 Thibauld Favre This program comes with ABSOLUTELY NO WARRANTY; for details type 'show w'. This is free software, and you are welcome to redistribute it under certain conditions; type 'show c' for details.

The hypothetical commands 'show w' and 'show c' should show the appropriate parts of the General Public License. Of course, your program's commands might be different; for a GUI interface, you would use an "about box".

You should also get your employer (if you work as a programmer) or school,

if any, to sign a "copyright disclaimer" for the program, if necessary. For more information on this, and how to apply and follow the GNU GPL, see <http://www.gnu.org/licenses/>.

The GNU General Public License does not permit incorporating your program

into proprietary programs. If your program is a subroutine library, you may consider it more useful to permit linking proprietary applications with the library. If this is what you want to do, use the GNU Lesser General Public License instead of this License. But first, please read <http://www.gnu.org/philosophy/why-not-lgpl.html>.

2.5 Help

Please contact Abdelrahman Hosny at abdelrahman.hosny@hotmail.com

CHAPTER 3

Need Help?

Please contact Abdelrahman Hosny at abdelrahman.hosny@ieee.org

CHAPTER 4

Authors

Abdelrahman Hosny, Paola Vera-Licona, Reinhard Laubenbacher & Thibauld Favre