Supercomputing InfiniBand Fabric Analysis

Todd Yoder

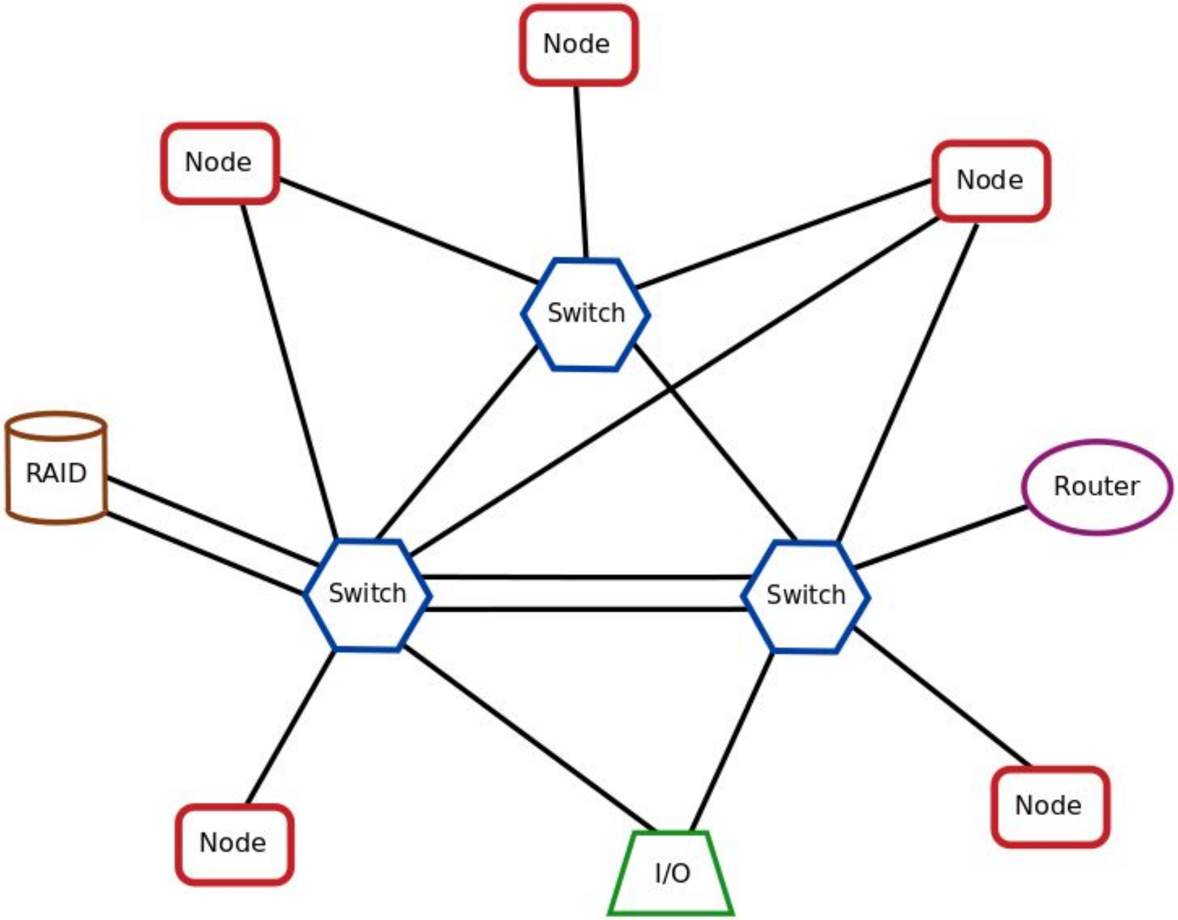
COLORADOSCHOOLOFMINES
EARTH • ENERGY • ENVIRONMENT

SIParCS
Summer Internships in Parallel Computational Science
BOULDER • CO

National Center for Atmospheric Research

August 3, 2018

Introduction

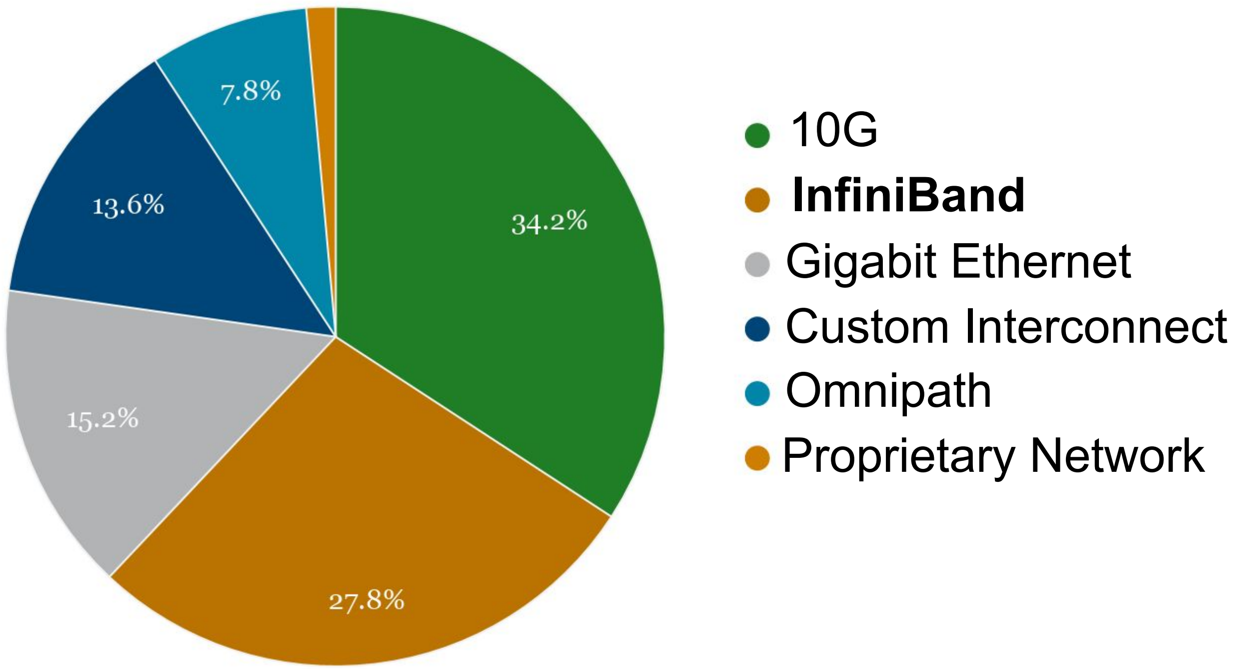


Simple Supercomputer Fabric

Introduction

InfiniBand is a computer-networking communications standard for high-performance computing.

Interconnect Family System Share

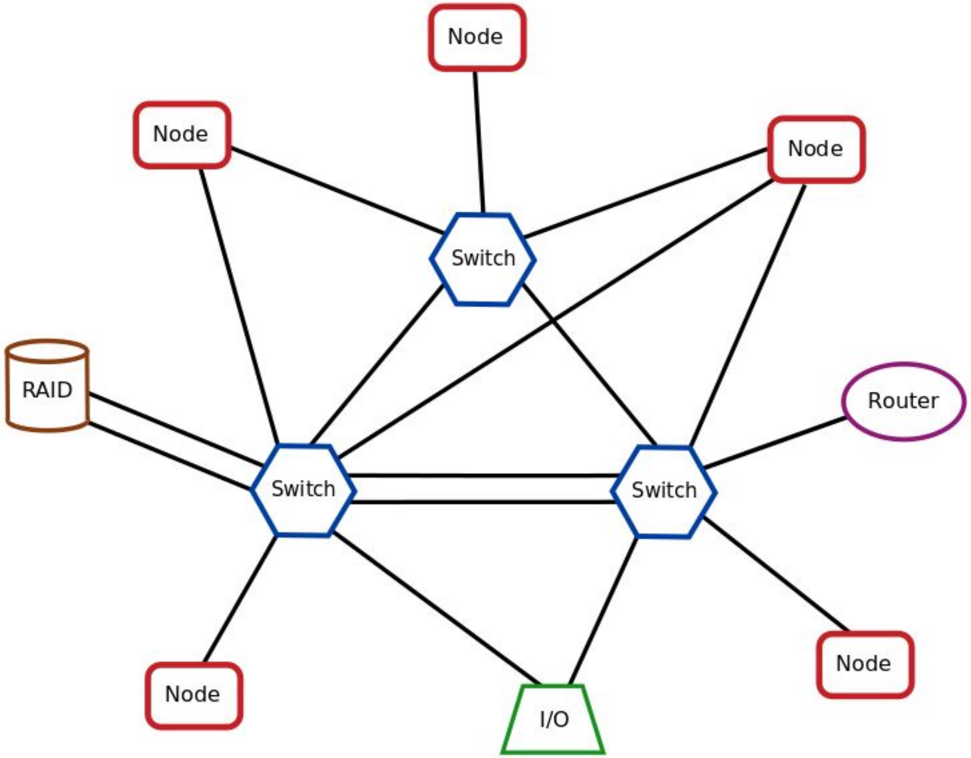


Interconnects used by the top 500 supercomputers[1]

Supercomputing InfiniBand Fabric Analysis

Goal

Develop software tools which analyze basic Graph Theory properties of an InfiniBand graph



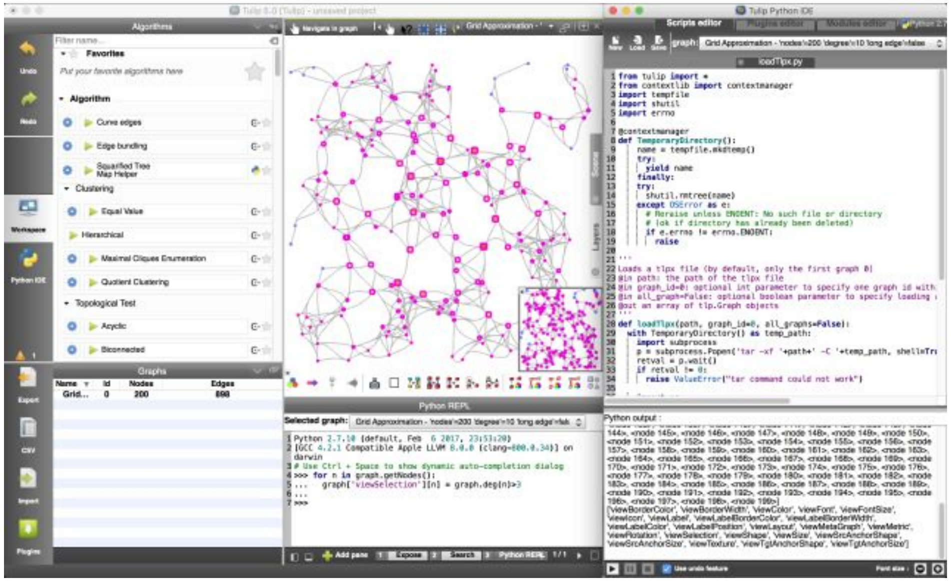
Simple Supercomputer Fabric

Tulip

Tulip is a free information visualization framework for analyzing and visualizing relational data. It can be extended with plugins to analyze specific problems.

Features:

- 3D visualizations
- Automatic drawing of graphs
- Automatic clustering of graphs
- Automatic Metric coloration of graphs
- Open Source
- Free
- Written in C++



graph: Grid Approximation - 'nodes'=200 'degree'=10 'long edge'=false

```
from tulip import *
from contextlib import contextmanager
import tempfile
import shutil
import errno

@contextmanager
def TemporaryDirectory():
    name = tempfile.mkdtemp()
    try:
      yield name
    finally:
    try:
      shutil.rmtree(name)
    except OSError as e:
      # Reraise unless ENOENT: No such file or directory
      # (ok if directory has already been deleted)
      if e.errno != errno.ENOENT:
        raise

'''
Loads a tlpx file (by default, only the first graph 0)
@in path: the path of the tlpx file
@in graph_id=0: optional int parameter to specify one graph id with
@in all_graph=False: optional boolean parameter to specify loading
@out an array of tlp.Graph objects
'''
def loadTlpx(path, graph_id=0, all_graphs=False):
  with TemporaryDirectory() as temp_path:
    import subprocess
    p = subprocess.Popen('tar -xf '+path+' -C '+temp_path, shell=Tr
    retval = p.wait()
    if retval != 0:
      raise ValueError("tar command could not work")
```

Selected graph: Grid Approximation - 'nodes'=200 'degree'=10 'long edge'=fals

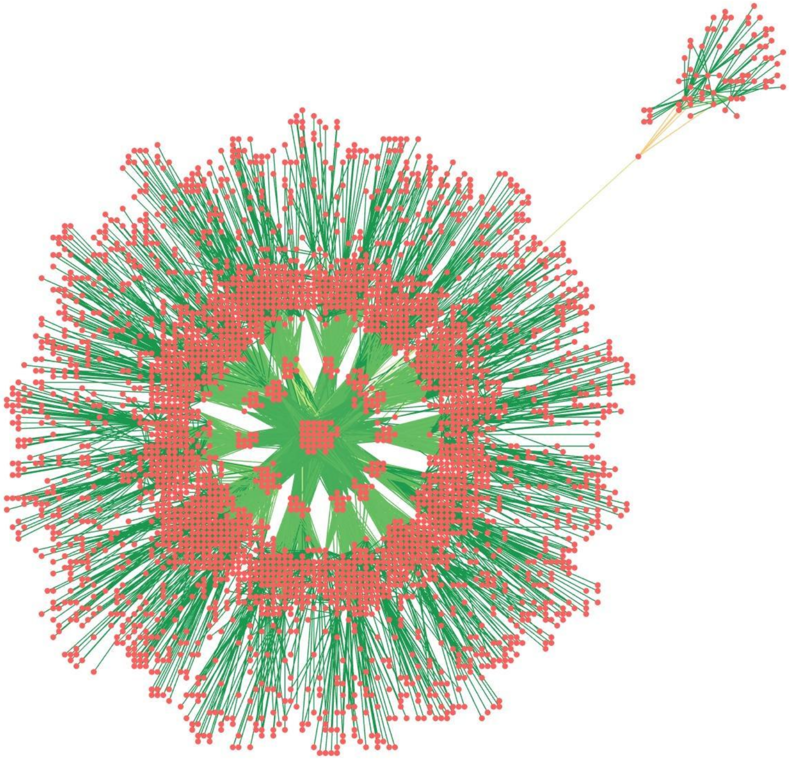
```
Python 2.7.10 (default, Feb  6 2017, 23:53:20)
[GCC 4.2.1 Compatible Apple LLVM 8.0.0 (clang-800.0.34)] on darwin
# Use Ctrl + Space to show dynamic auto-completion dialog
>>> for n in graph.getNodes():
...     graph['viewSelection'][n] = graph.deg(n)>3
...
>>>
```

Python output :

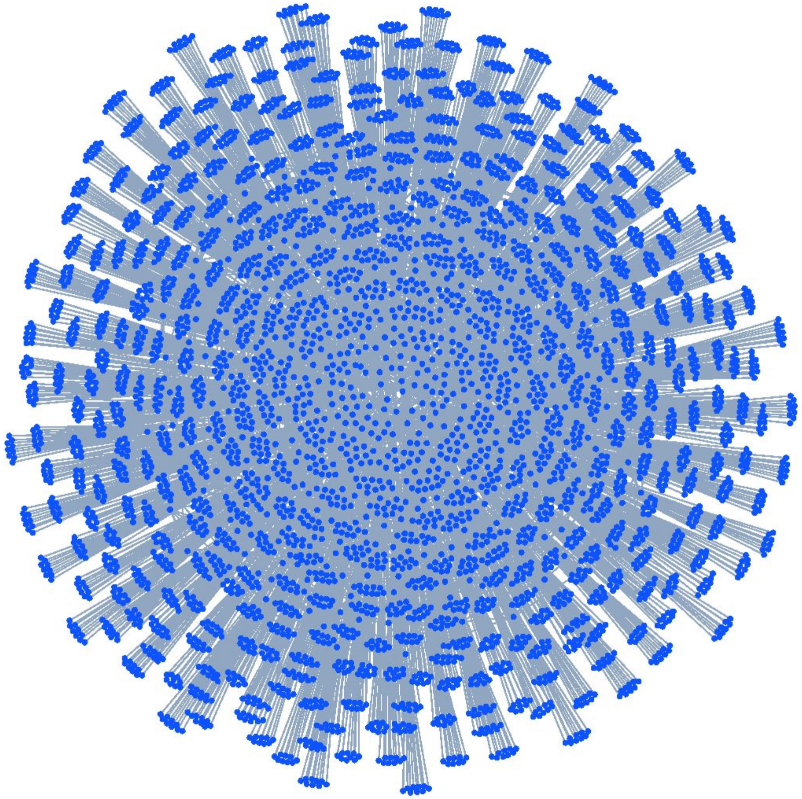
```
144>, <node 145>, <node 146>, <node 147>, <node 148>, <node 149>, <node 150>,
<node 151>, <node 152>, <node 153>, <node 154>, <node 155>, <node 156>, <node
157>, <node 158>, <node 159>, <node 160>, <node 161>, <node 162>, <node 163>,
<node 164>, <node 165>, <node 166>, <node 167>, <node 168>, <node 169>, <node
170>, <node 171>, <node 172>, <node 173>, <node 174>, <node 175>, <node 176>,
<node 177>, <node 178>, <node 179>, <node 180>, <node 181>, <node 182>, <node
183>, <node 184>, <node 185>, <node 186>, <node 187>, <node 188>, <node 189>,
<node 190>, <node 191>, <node 192>, <node 193>, <node 194>, <node 195>, <node
196>, <node 197>, <node 198>, <node 199>]
['viewBorderColor', 'viewBorderWidth', 'viewColor', 'viewFont', 'viewFontSize',
'viewIcon', 'viewLabel', 'viewLabelBorderColor', 'viewLabelBorderWidth',
'viewLabelColor', 'viewLabelPosition', 'viewLayout', 'viewMetaGraph', 'viewMetric',
'viewRotation', 'viewSelection', 'viewShape', 'viewSize', 'viewSrcAnchorShape',
'viewSrcAnchorSize', 'viewTexture', 'viewTgtAnchorShape', 'viewTgtAnchorSize']
```

Sample screenshot of Tulip's graphic user interface[2]

Tulip



NCAR's Yellowstone supercomputer, a full fat tree[3]



NCAR's Cheyenne supercomputer, a partial 9D enhanced hypercube

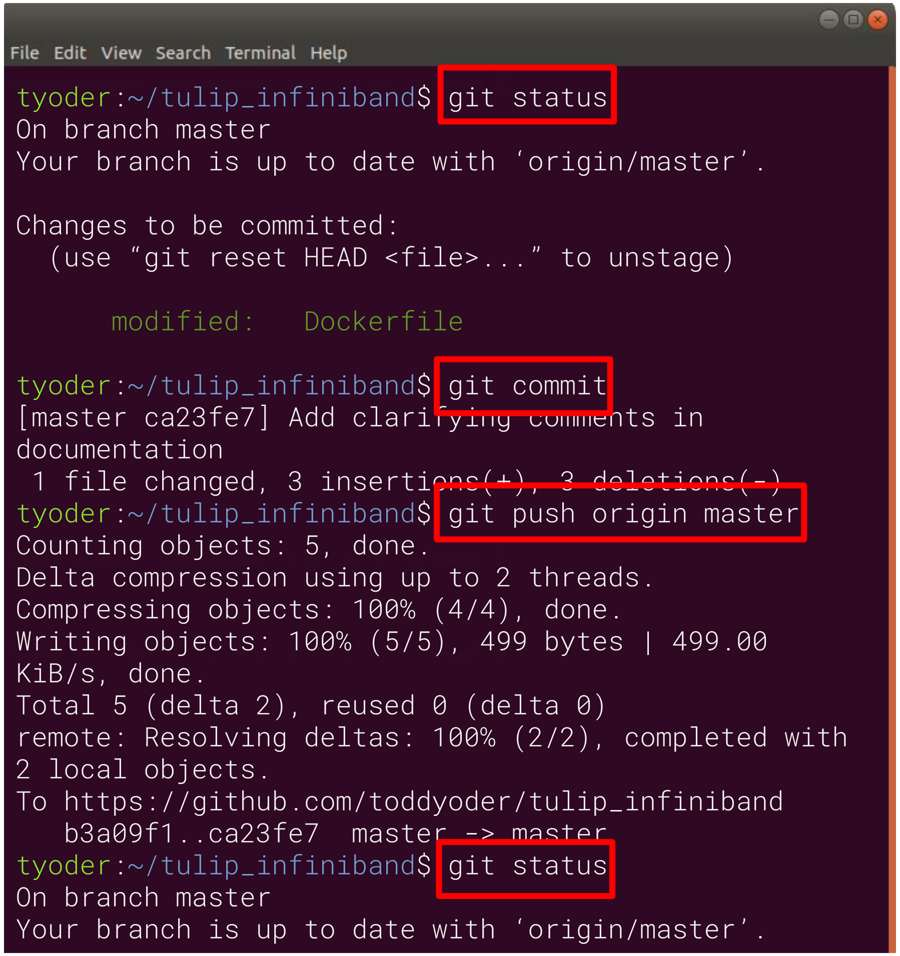
GitHub git

https://github.com/NCAR/tulip_infiniband

GitHub: collaboration manager and web-based hosting service for git

git: version control

GitHub branches in a repository[4]



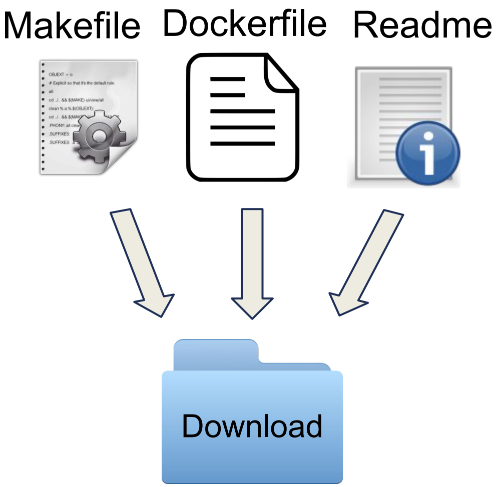
```
tyoder:~/tulip_infiniband$ git status
On branch master
Your branch is up to date with 'origin/master'.

Changes to be committed:
  (use "git reset HEAD <file>..." to unstage)

        modified:   Dockerfile

tyoder:~/tulip_infiniband$ git commit
[master ca23fe7] Add clarifying comments in
documentation
 1 file changed, 3 insertions(+), 3 deletions(-)
tyoder:~/tulip_infiniband$ git push origin master
Counting objects: 5, done.
Delta compression using up to 2 threads.
Compressing objects: 100% (4/4), done.
Writing objects: 100% (5/5), 499 bytes | 499.00
KiB/s, done.
Total 5 (delta 2), reused 0 (delta 0)
remote: Resolving deltas: 100% (2/2), completed with
2 local objects.
To https://github.com/toddyoder/tulip_infiniband
   b3a09f1..ca23fe7  master -> master
tyoder:~/tulip_infiniband$ git status
On branch master
Your branch is up to date with 'origin/master'.
```

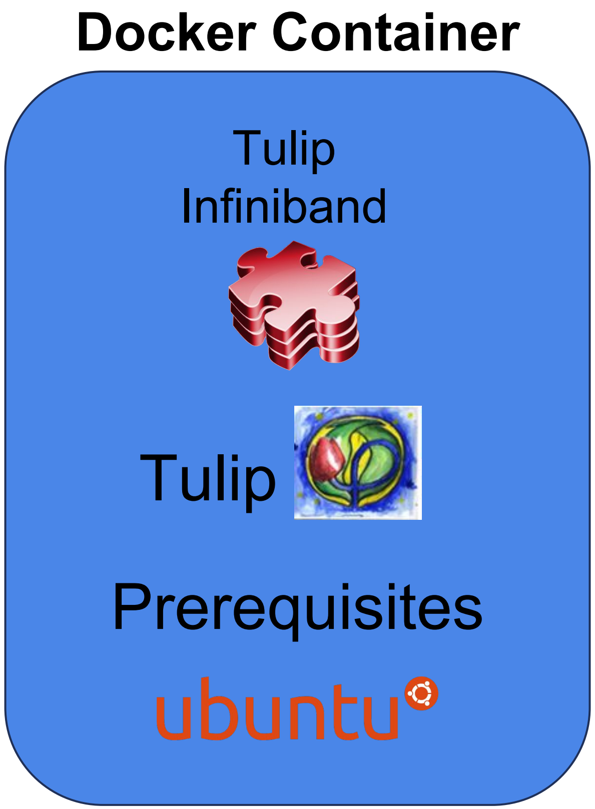
Docker



1. Install Docker, gcc

2. Download Tulip Infiniband docker folder

3. $ make



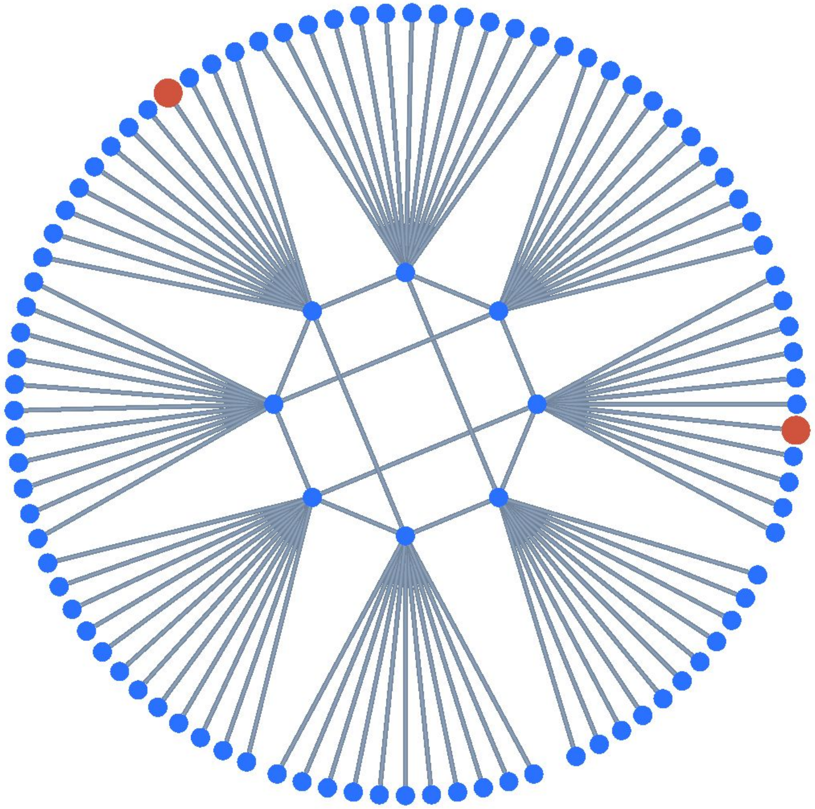
Plugins Developed

Random Nodes

Selects two random nodes on the graph

Specific Application:

Used by other plugins



Laramie: a 3D hypercube test and research supercomputer at NCAR

Plugins Developed

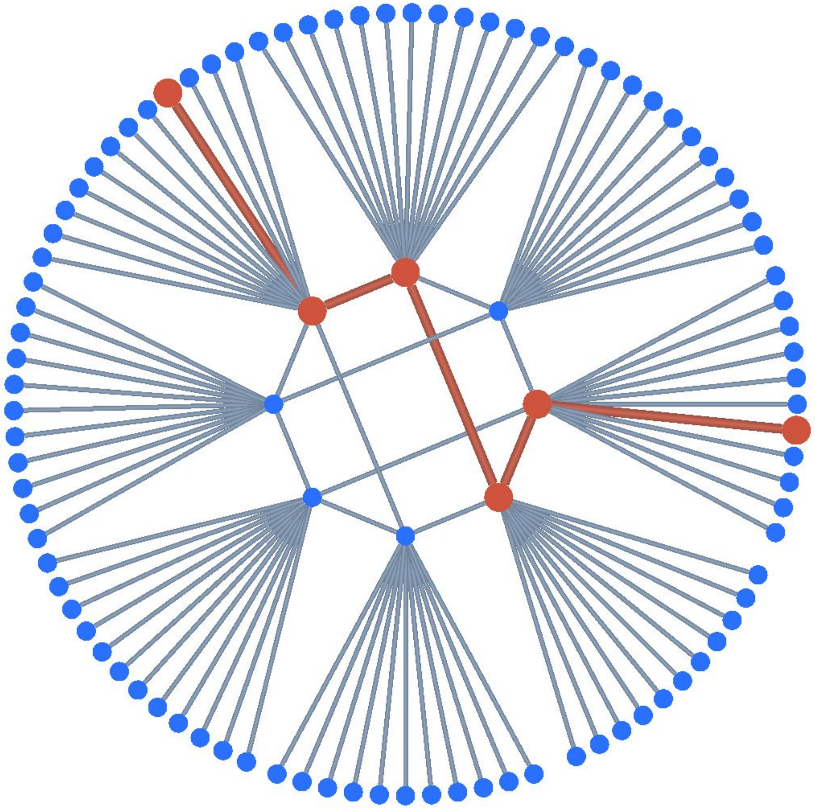
Shortest Path

Applies Dijkstra’s Algorithm to one of the nodes.

Selects a shortest path between the nodes

Specific Application:

- Find routes nodes ought to use to communicate.
- Compare optimal routes with actual routes



A shortest path between two nodes on Laramie

Plugins Developed

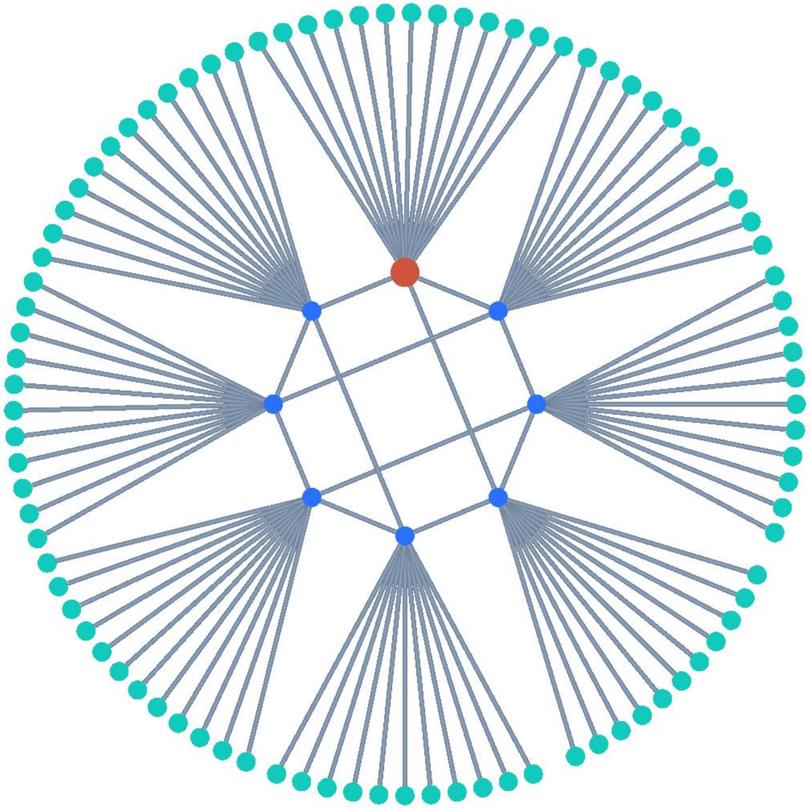
Min Degree and **Max Degree**

Prints smallest and largest node degrees, respectively

Selects corresponding nodes and prints their node IDs

Specific Application:

- Determine where network congestion is likely to occur
- Minimize number of cables in supercomputer while maintaining communication capabilities



Largest degree: 42, Smallest degree: 2

Plugins Developed

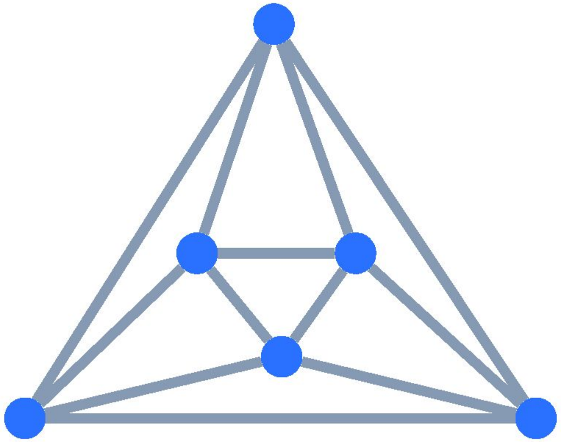
Regularity Test

Regular Graph: all nodes have the same degree

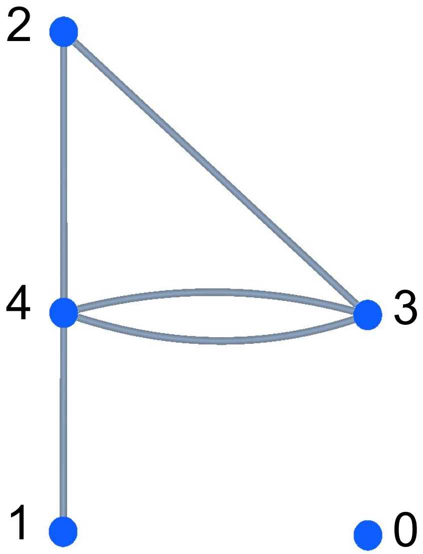
Irregular Graph: each node has a unique degree

Specific Application:

Determine if switches are not symmetric



A regular graph.
Each node has degree 4



An irregular graph with degrees labeled

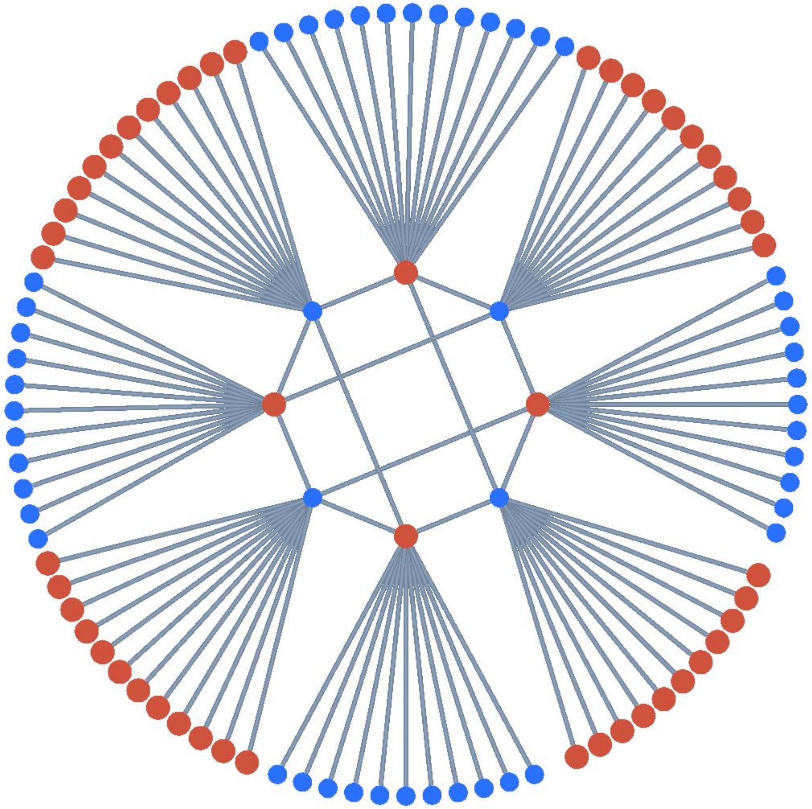
Plugins Developed

Bipartite Test

Bipartite Graph: The nodes can be partitions into two subsets such that every edge connects the two subsets

Specific Application:

Enables straightforward full-fabric bandwidth testing



Laramie is bipartite

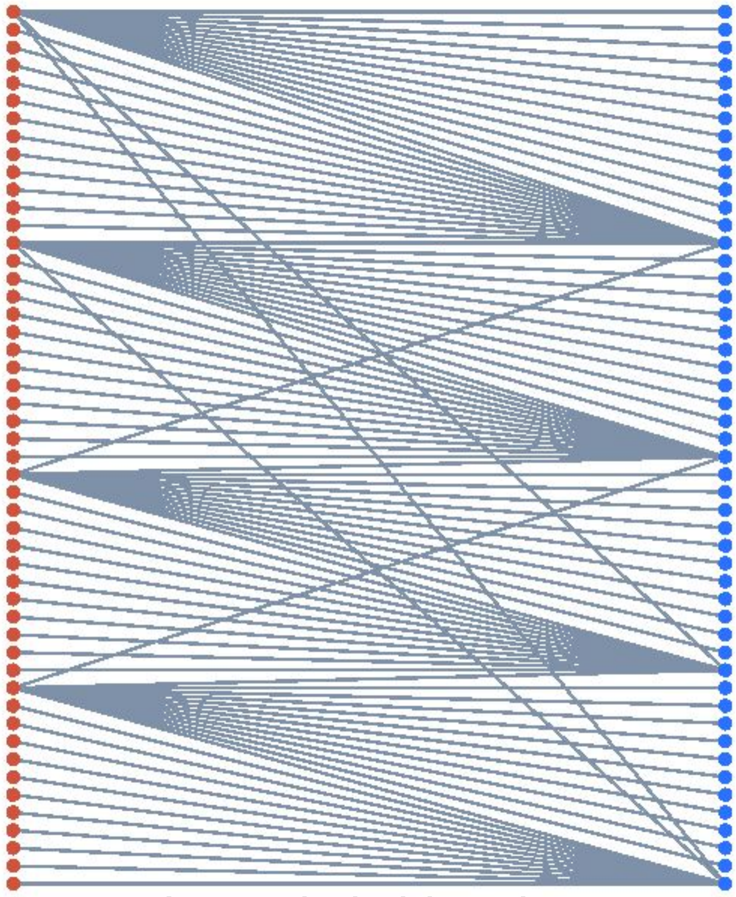
Plugins Developed

Bipartite Test

Bipartite Graph: The nodes can be partitions into two subsets such that every edge connects the two subsets

Specific Application:

Enables straightforward full-fabric bandwidth testing



Laramie is bipartite

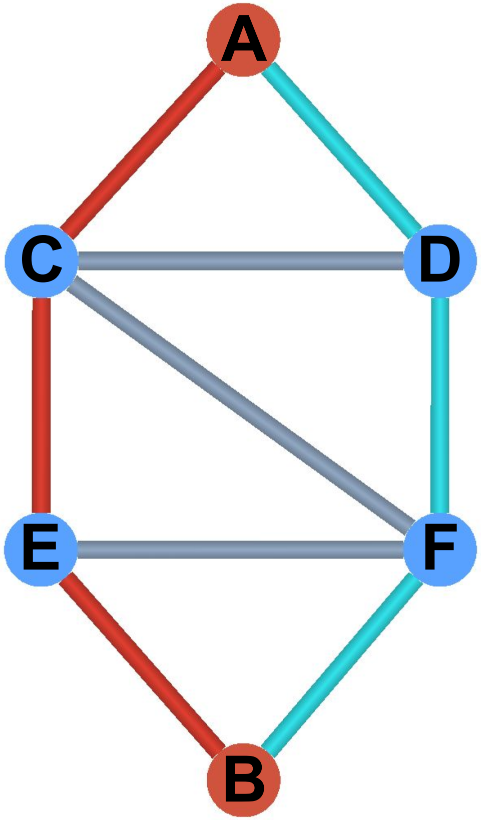
Plugins Developed

Geodesic Test

Geodesic Path: path of shortest length between two nodes

Specific Application:

- Fabrics need redundancy. It’s useful to check that more than one optimal paths exist between nodes
- Helps check for excessive cables



Three geodesic paths from A to B:
red, blue, and ACFB

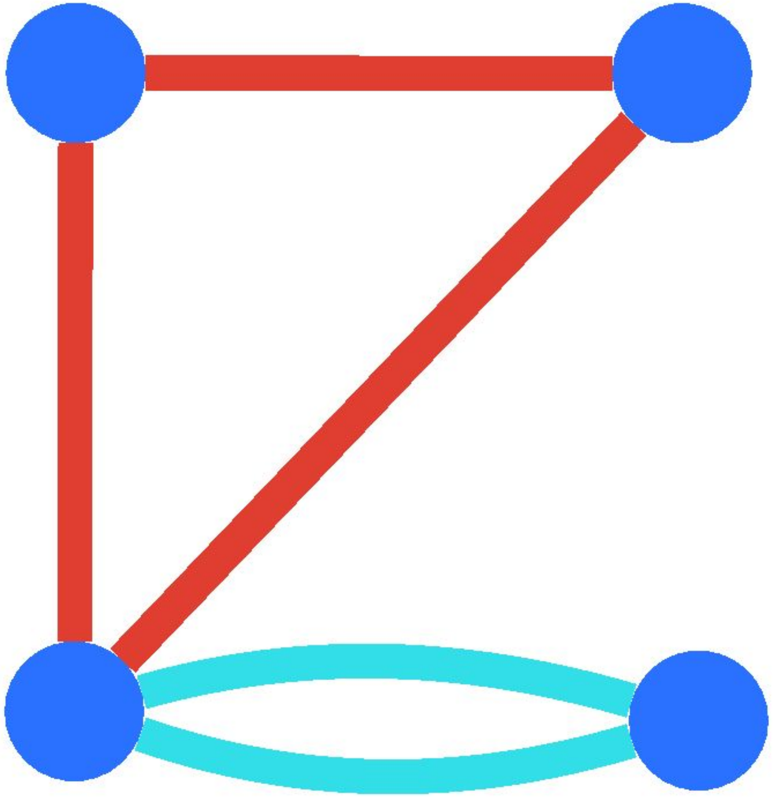
Plugins Developed

Node On Cycle Test

Determines if the selected node lies on a cycle

Specific Application:

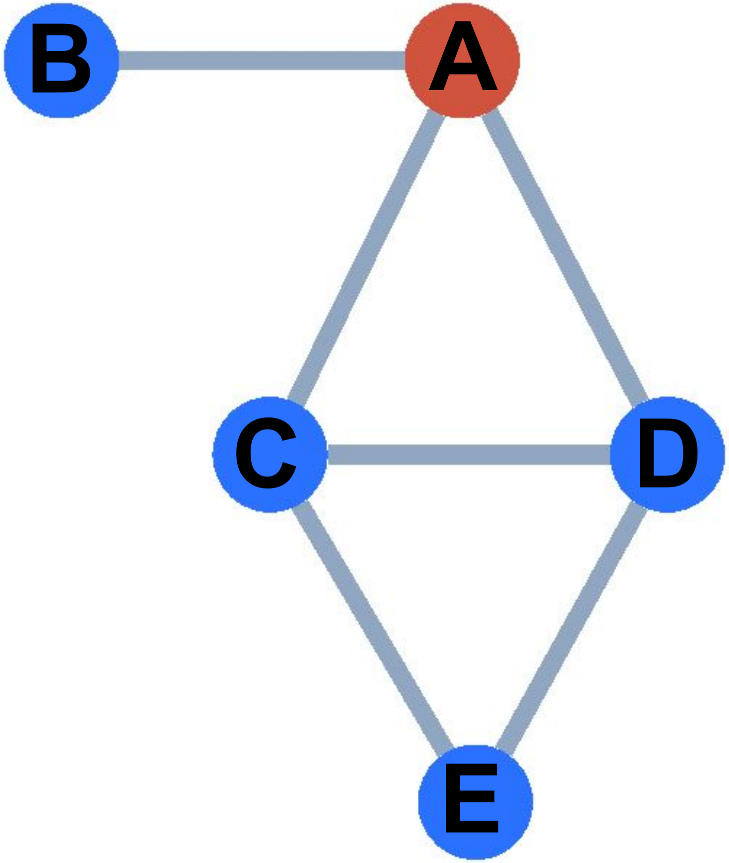
Multicast communications need to be aware of cycles to guard against inefficiencies and infinite loops



Graph with two cycles

Plugins Developed

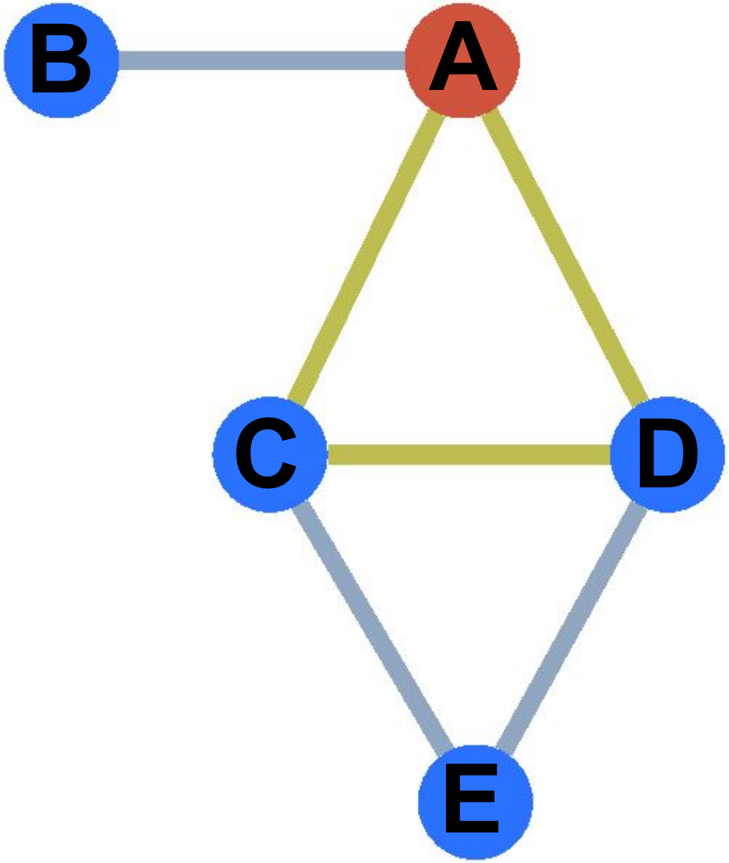
Node On Cycle Test



Multicast: Send message to multiple nodes, they store and pass on the message

Plugins Developed

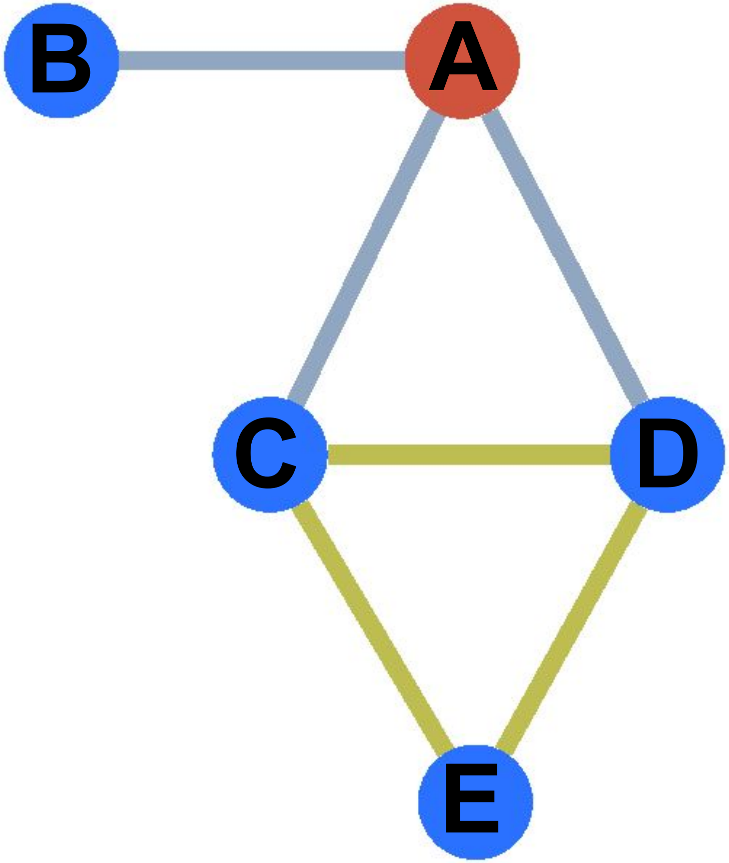
Node On Cycle Test



Route ACD is inefficient

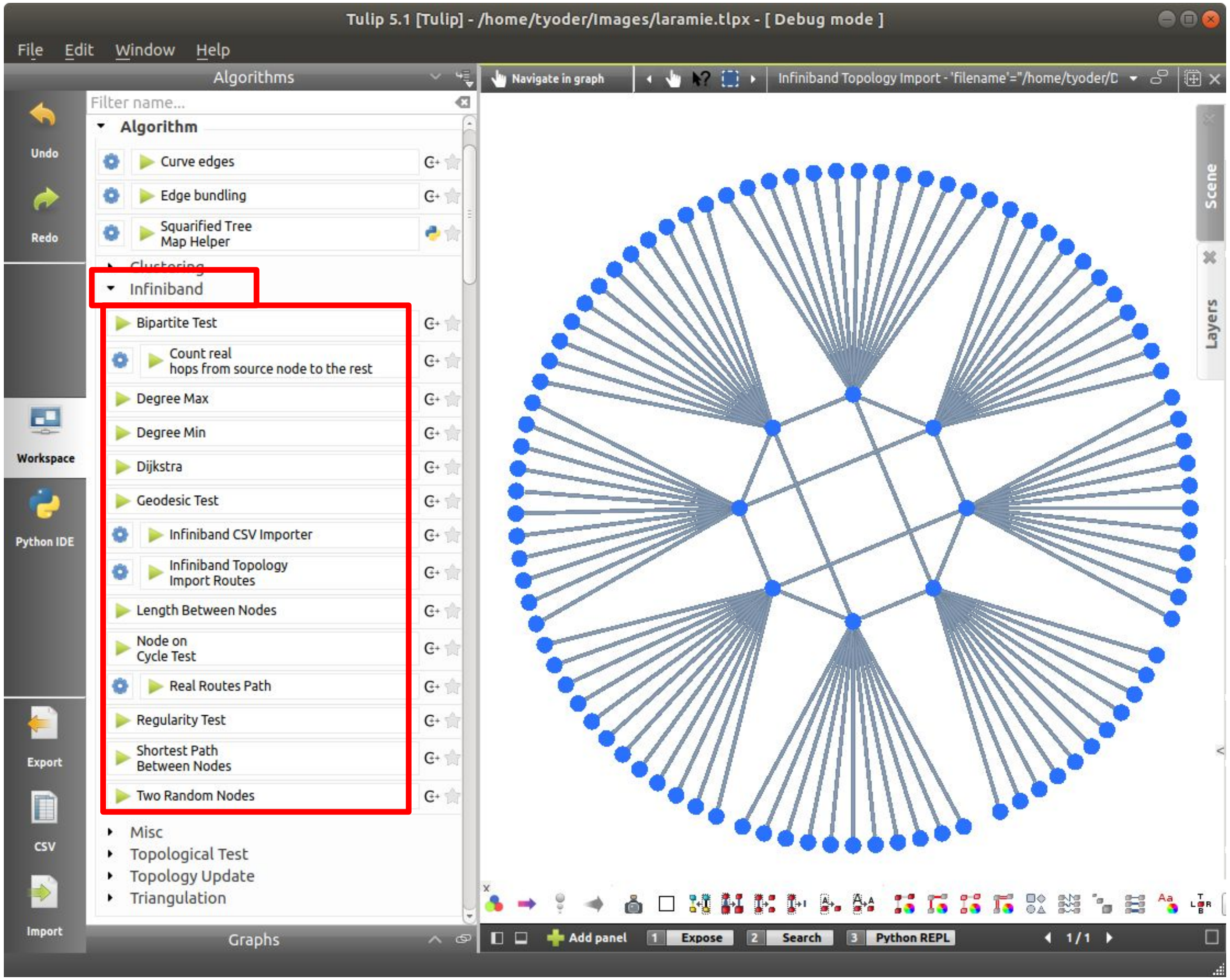
Plugins Developed

Node On Cycle Test

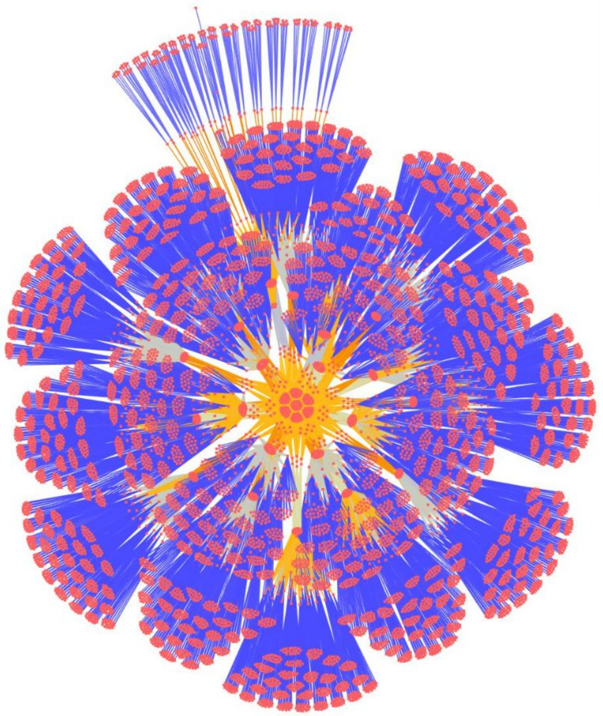


Route CED is an infinite loop!

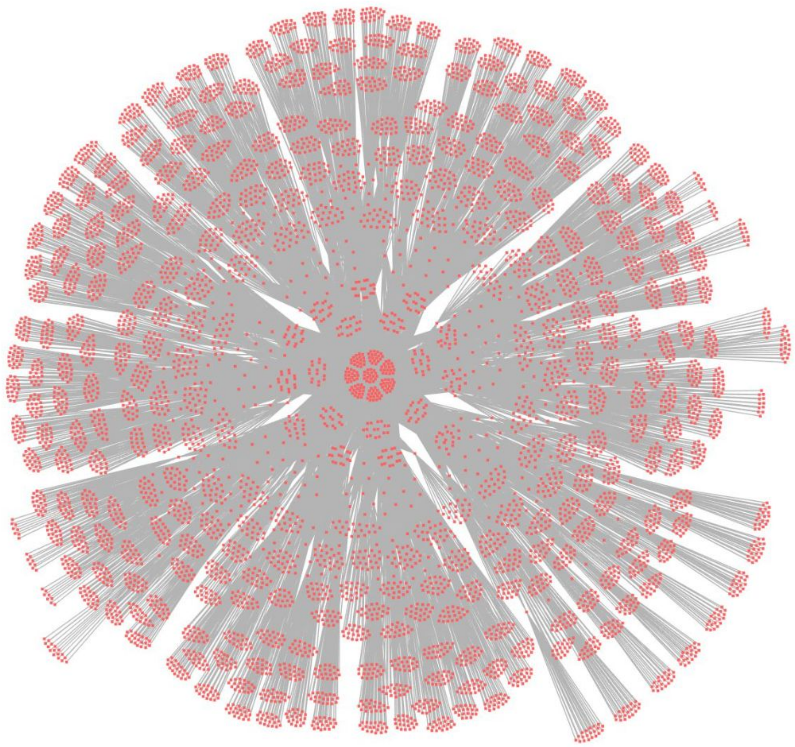
Using Tulip Infiniband



Using Tulip Infiniband



SuperMUC, a supercomputer operated by Leibniz Supercomputing Center



Stampede, a supercomputer operated by Texas Advanced Computing Center until 2017

Conclusions

Tulip Infiniband can help supercomputer development teams such as SSG make more informed decisions for upgrades, and it provides basic tools for maintenance and performance optimization.

Future Work

- Write plugins for other graph theory properties
- Convert Dockerfile to Charliecloud or Singularity
- Write plugin which generates a summary of the graph by calling other plugins

Acknowledgements

- Auber, D., & Mary, P. (2018). Tulip (Version 5.2) [Computer software]. Bordeaux, France: LaBRI, University of Bordeaux I.
- Chartrand, G., & Zhang, P. (2005). *Introduction to graph theory*. Boston: McGraw-Hill Higher Education.
- Futral, W. T. (2002). *InfiniBand architecture development and deployment: A strategic guide to server I/O solutions*. Hillsboro, OR: Intel Press.

Special thanks to:

Mentors:

- Nate Rini, Tom Kleespies

SIParCS Team:

- AJ Lauer, Rich Loft, Elliot Foust, Jenna Preston

Support:

- Shilo Hall, Ben Matthews

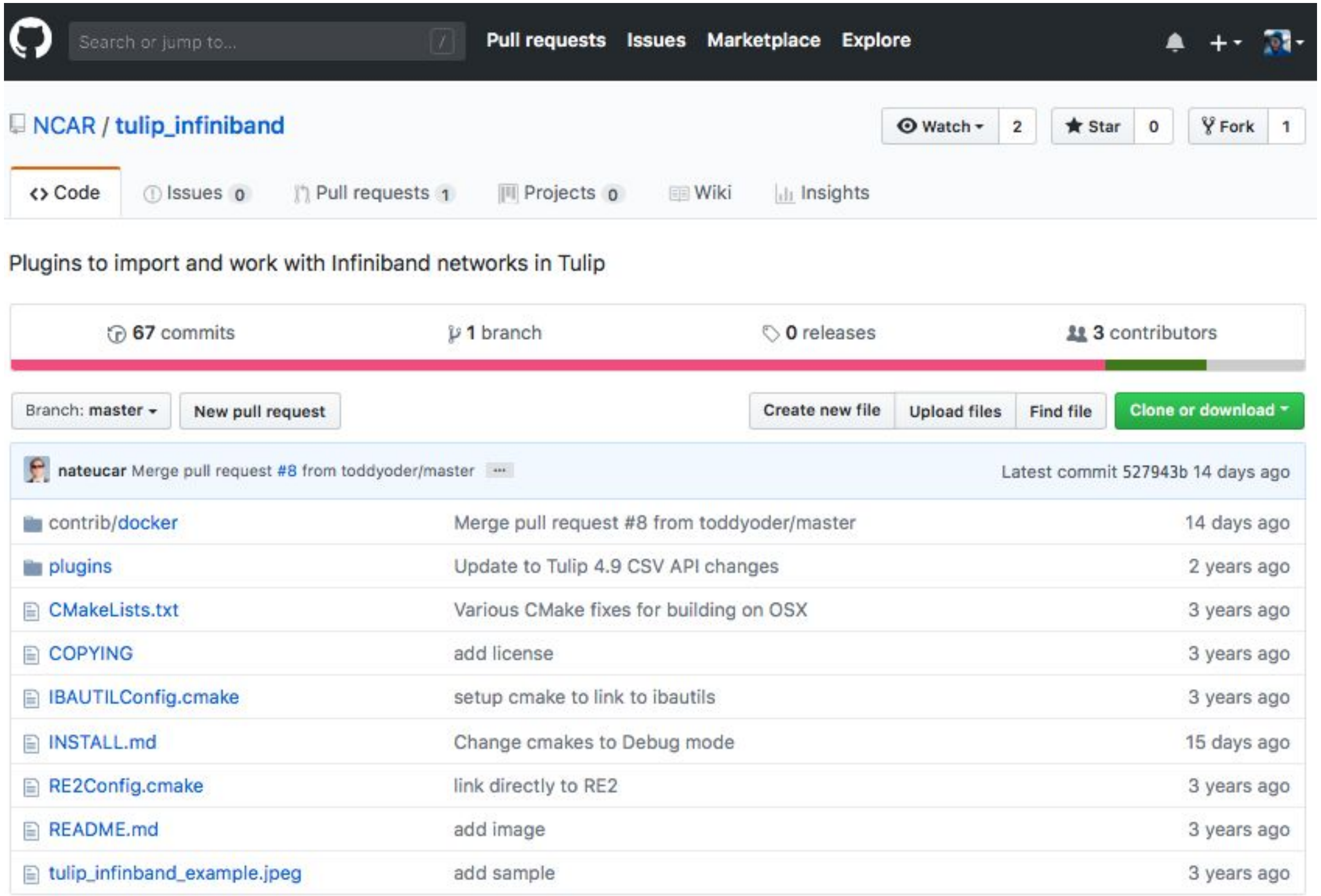
Overseeing Organizations:

- NSF
- UCAR
- NCAR

The National Center for Atmospheric Research is sponsored by the National Science Foundation. Any opinions, findings and conclusions or recommendations expressed in this publication are those of the author(s) and do not necessarily reflect the views of the National Science Foundation.

Questions?

https://github.com/NCAR/tulip_infiniband



NCAR / tulip_infiniband

Plugins to import and work with Infiniband networks in Tulip

67 commits | 1 branch | 0 releases | 3 contributors

nateucar Merge pull request #8 from toddyoder/master ... Latest commit 527943b 14 days ago

contrib/docker	Merge pull request #8 from toddyoder/master	14 days ago
plugins	Update to Tulip 4.9 CSV API changes	2 years ago
CMakeLists.txt	Various CMake fixes for building on OSX	3 years ago
COPYING	add license	3 years ago
IBAUTILConfig.cmake	setup cmake to link to ibautils	3 years ago
INSTALL.md	Change cmakes to Debug mode	15 days ago
RE2Config.cmake	link directly to RE2	3 years ago
README.md	add image	3 years ago
tulip_infinband_example.jpeg	add sample	3 years ago

Backup Slides

Compatibility

Mac doesn’t play nice with Graphical User Interfaces in Docker.

XQuartz bridges the gap to provide a GUI through the IP address.

Linux

1. Install Docker, gcc
2. Download Tulip Infiniband Docker folder
3. $ make

Mac

1. Install Docker, gcc, XQuartz
2. Download Tulip Infiniband Docker folder
3. $ make

The Dockerfile

1 **Load Ubuntu image**

Docker provides images with many popular operating systems

2 **Install Prerequisites**

Tulip and the plugins depend on about two dozen libraries

3 **Install Tulip**

Tulip is available at https://github.com/Tulip-Dev/tulip

4 **Install libibautils**

Imports InfiniBand fabric into Tulip. Developed at NCAR

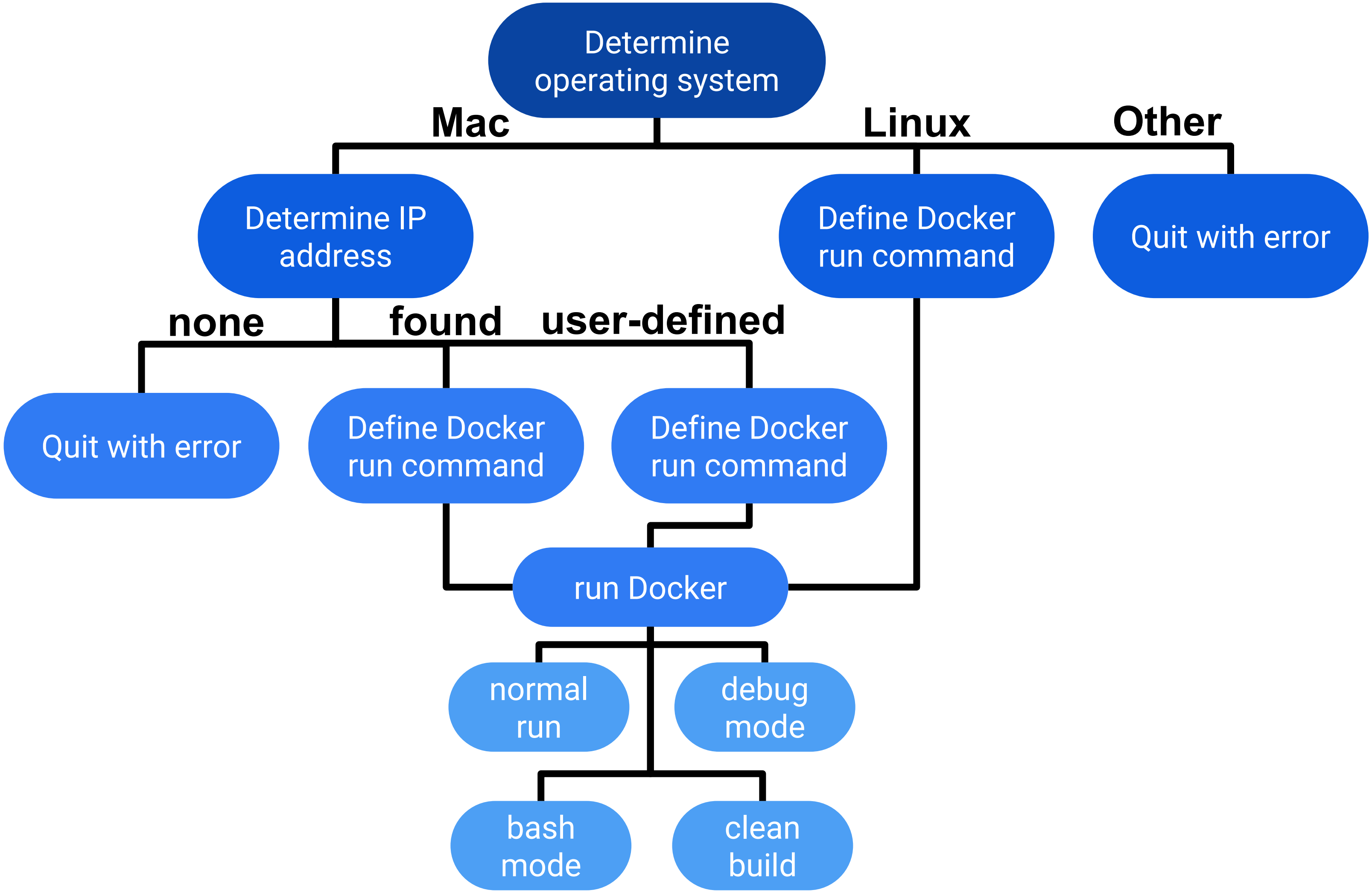
5 **Install Tulip Infiniband**

Analysis plugins developed for InfiniBand

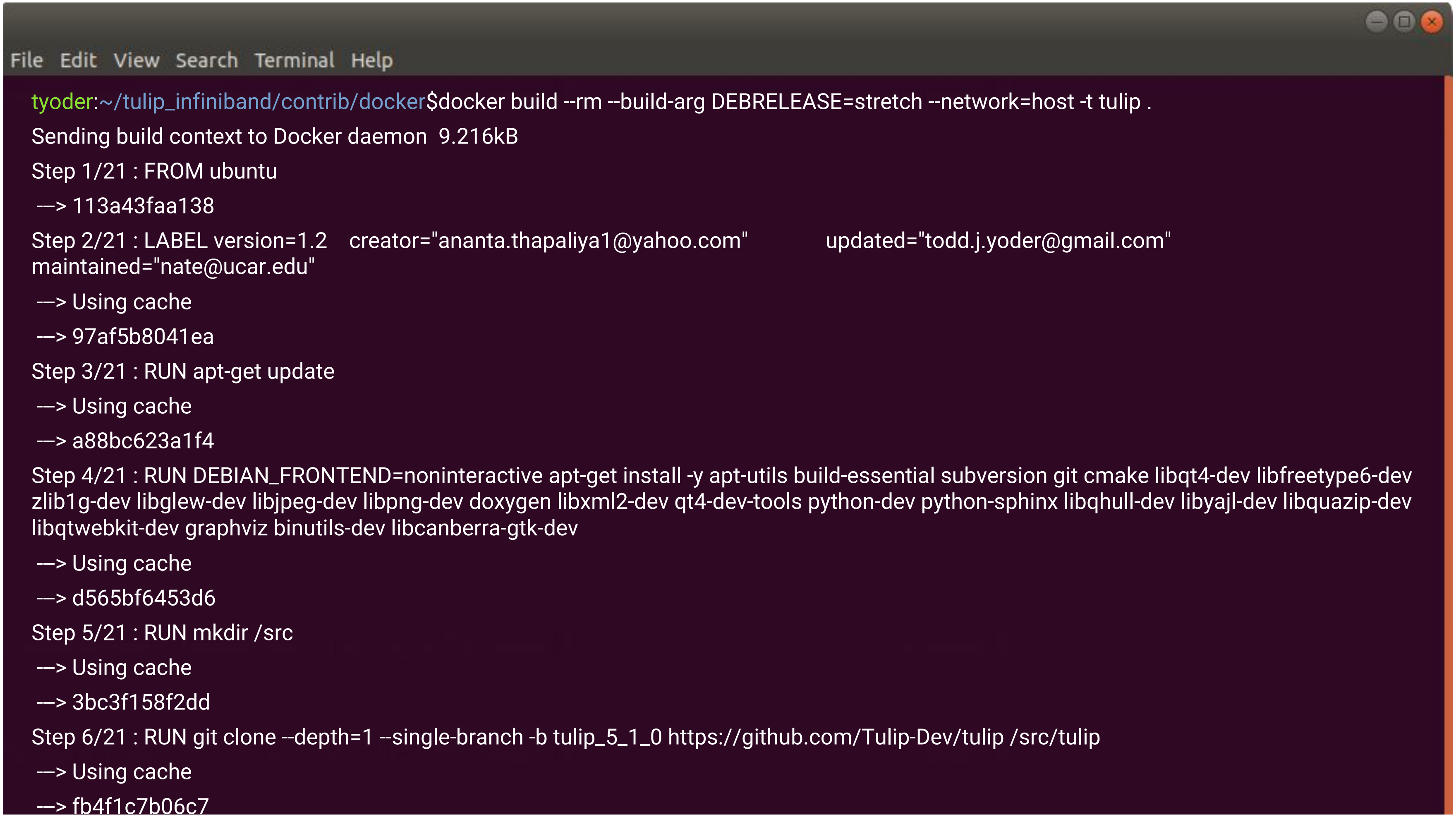
6 **Run Tulip**

Tulip launches with libibautils and Tulip Infiniband plugins

The Makefile

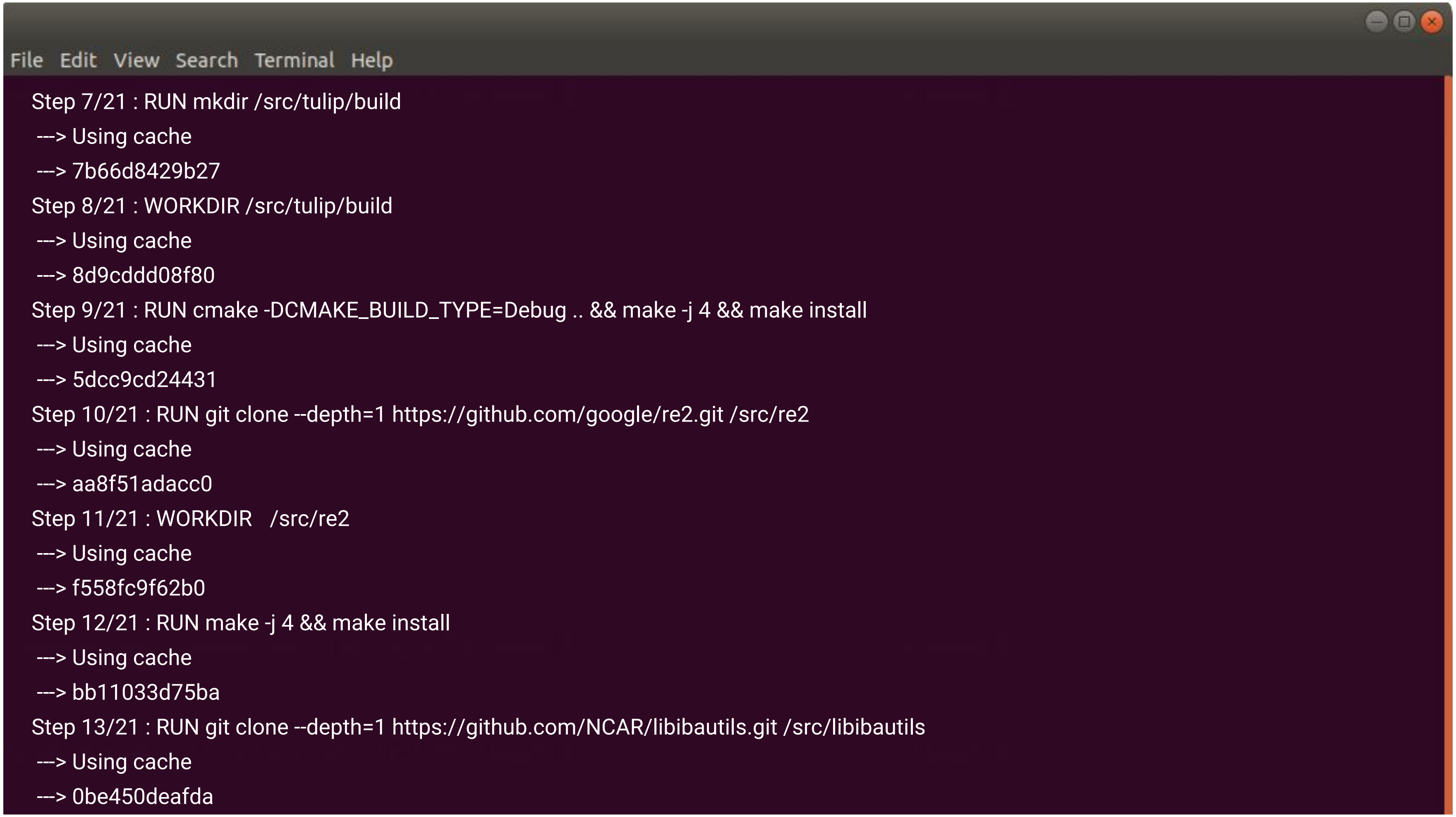


Dockerfile



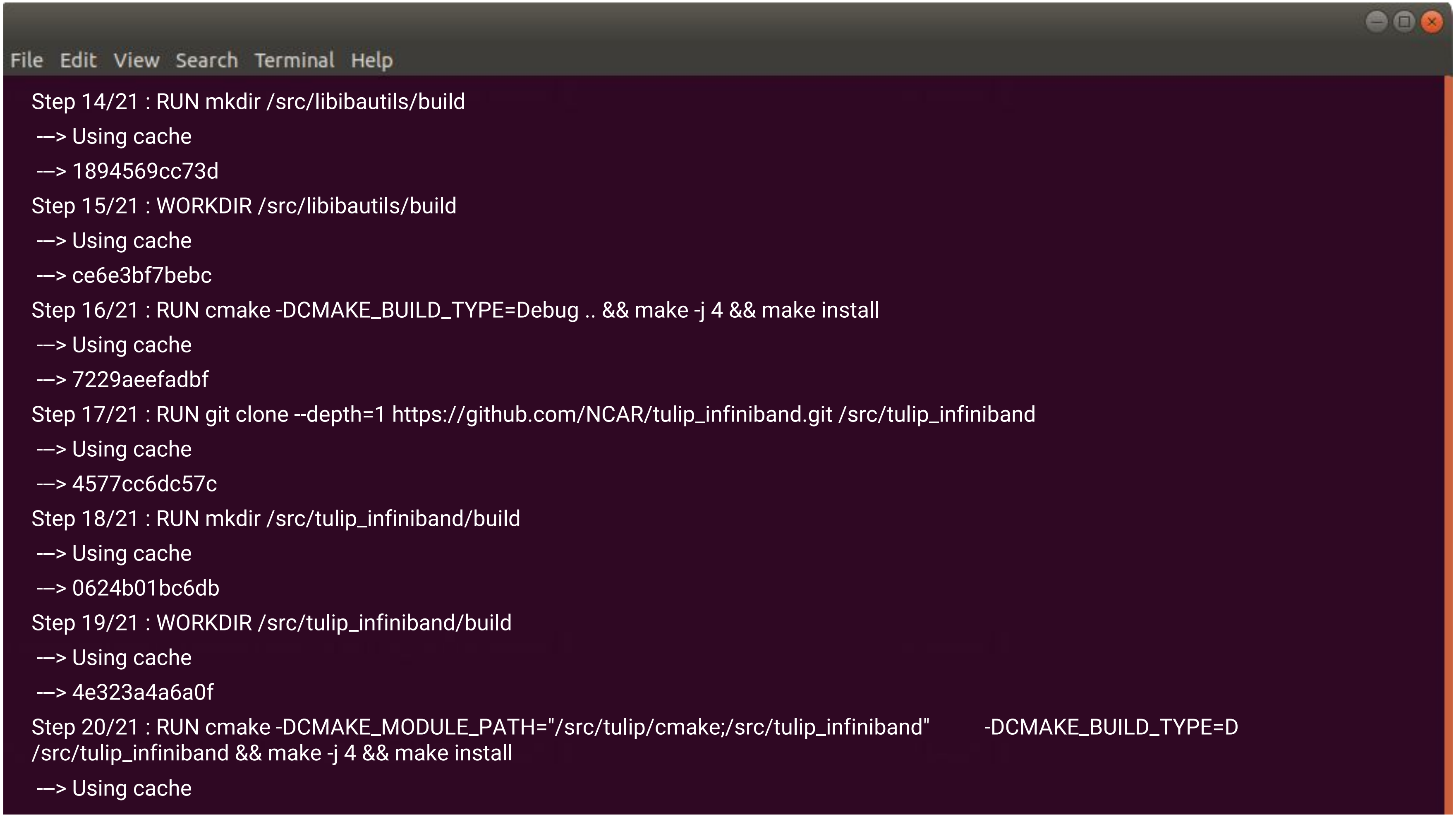
```
tyoder:~/tulip_infiniband/contrib/docker$docker build --rm --build-arg DEBRELEASE=stretch --network=host -t tulip .
Sending build context to Docker daemon  9.216kB
Step 1/21 : FROM ubuntu
 ---> 113a43faa138
Step 2/21 : LABEL version=1.2    creator="ananta.thapaliya1@yahoo.com"            updated="todd.j.yoder@gmail.com"
maintained="nate@ucar.edu"
 ---> Using cache
 ---> 97af5b8041ea
Step 3/21 : RUN apt-get update
 ---> Using cache
 ---> a88bc623a1f4
Step 4/21 : RUN DEBIAN_FRONTEND=noninteractive apt-get install -y apt-utils build-essential subversion git cmake libqt4-dev libfreetype6-dev
zlib1g-dev libglew-dev libjpeg-dev libpng-dev doxygen libxml2-dev qt4-dev-tools python-dev python-sphinx libqhull-dev libyajl-dev libquazip-dev
libqtwebkit-dev graphviz binutils-dev libcanberra-gtk-dev
 ---> Using cache
 ---> d565bf6453d6
Step 5/21 : RUN mkdir /src
 ---> Using cache
 ---> 3bc3f158f2dd
Step 6/21 : RUN git clone --depth=1 --single-branch -b tulip_5_1_0 https://github.com/Tulip-Dev/tulip /src/tulip
 ---> Using cache
 ---> fb4f1c7b06c7
```

Dockerfile



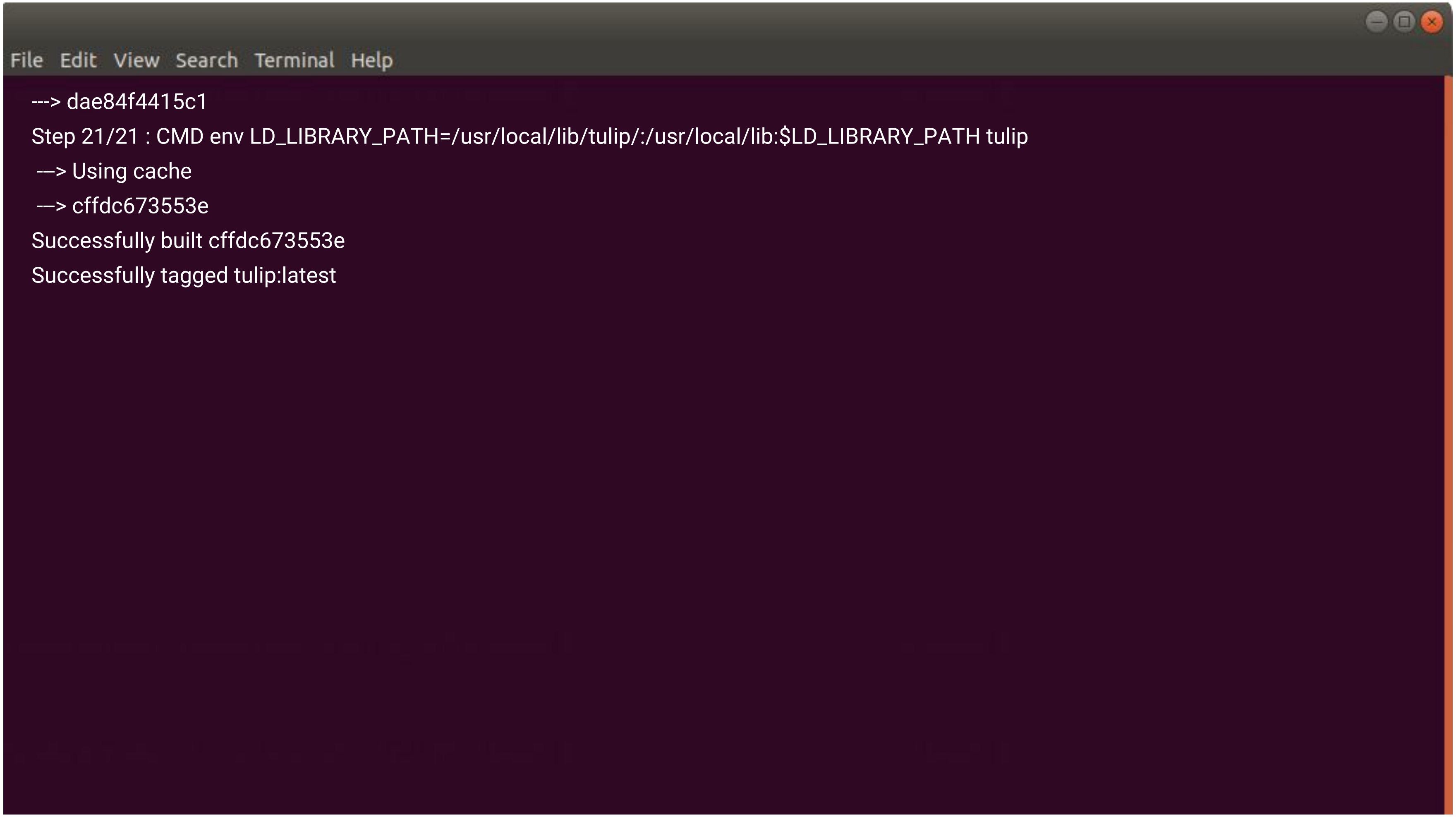
```
Step 7/21 : RUN mkdir /src/tulip/build
 ---> Using cache
 ---> 7b66d8429b27
Step 8/21 : WORKDIR /src/tulip/build
 ---> Using cache
 ---> 8d9cddd08f80
Step 9/21 : RUN cmake -DCMAKE_BUILD_TYPE=Debug .. && make -j 4 && make install
 ---> Using cache
 ---> 5dcc9cd24431
Step 10/21 : RUN git clone --depth=1 https://github.com/google/re2.git /src/re2
 ---> Using cache
 ---> aa8f51adacc0
Step 11/21 : WORKDIR   /src/re2
 ---> Using cache
 ---> f558fc9f62b0
Step 12/21 : RUN make -j 4 && make install
 ---> Using cache
 ---> bb11033d75ba
Step 13/21 : RUN git clone --depth=1 https://github.com/NCAR/libibautils.git /src/libibautils
 ---> Using cache
 ---> 0be450deafda
```

Dockerfile



```
Step 14/21 : RUN mkdir /src/libibautils/build
 ---> Using cache
 ---> 1894569cc73d
Step 15/21 : WORKDIR /src/libibautils/build
 ---> Using cache
 ---> ce6e3bf7bebc
Step 16/21 : RUN cmake -DCMAKE_BUILD_TYPE=Debug .. && make -j 4 && make install
 ---> Using cache
 ---> 7229aeefadbf
Step 17/21 : RUN git clone --depth=1 https://github.com/NCAR/tulip_infiniband.git /src/tulip_infiniband
 ---> Using cache
 ---> 4577cc6dc57c
Step 18/21 : RUN mkdir /src/tulip_infiniband/build
 ---> Using cache
 ---> 0624b01bc6db
Step 19/21 : WORKDIR /src/tulip_infiniband/build
 ---> Using cache
 ---> 4e323a4a6a0f
Step 20/21 : RUN cmake -DCMAKE_MODULE_PATH="/src/tulip/cmake;/src/tulip_infiniband"        -DCMAKE_BUILD_TYPE=D
/src/tulip_infiniband && make -j 4 && make install
 ---> Using cache
```

Dockerfile



```
---> dae84f4415c1
Step 21/21 : CMD env LD_LIBRARY_PATH=/usr/local/lib/tulip/:/usr/local/lib:$LD_LIBRARY_PATH tulip
 ---> Using cache
 ---> cffdc673553e
Successfully built cffdc673553e
Successfully tagged tulip:latest
```

Plugin Algorithms

Bipartite Test

1. Set selected node as src
2. Place src in Group A
3. Place all neighbors of src in Group B
4. Place the neighbors' neighbors in Group A
5. Continue until all nodes are classified
6. Not bipartite if a node belongs to both groups

Geodesic Test

1. Verify selected edges form a path
2. Count number of edges selected
3. Call Shortest Path plugin to get length between the end nodes
4. Compare lengths

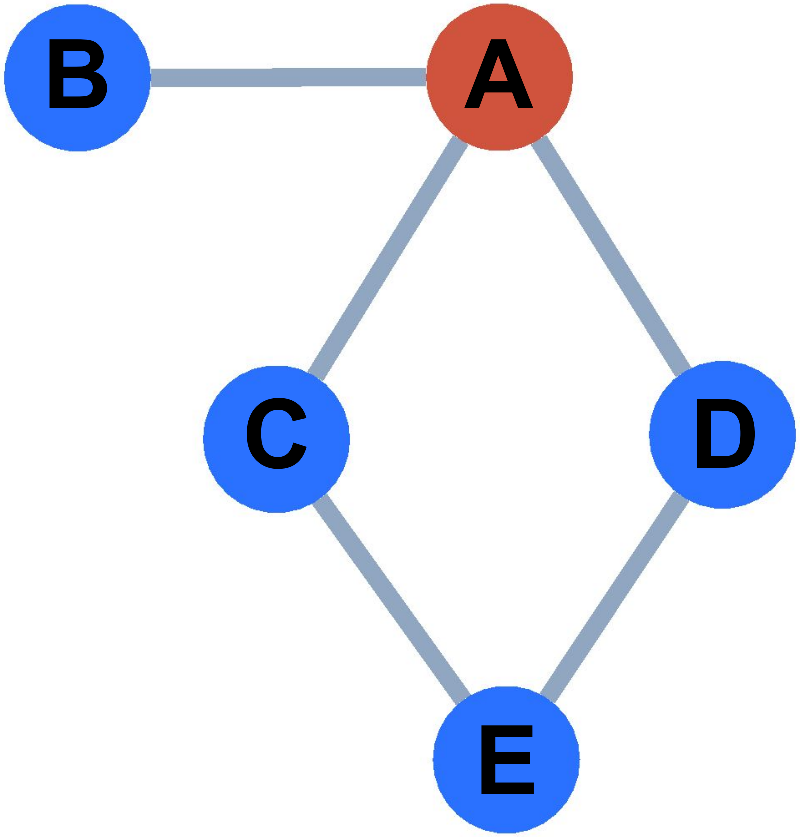
Plugin Algorithms

Node On Cycle Test

1. Store selected node as src
2. Check src degree. False if degree < 2
3. Check for self-loops
4. Check for two edges connecting src to the same neighbor
5. For node N, beginning with src,
   a. Store N as parent of all parentless neighbors, unless src
   b. If a neighbor already has a parent, src is on a cycle if the paths to N and its neighbor are disjoint except for src
   c. Add N's neighbors to queue to be considered

Plugins Developed

Node On Cycle Test

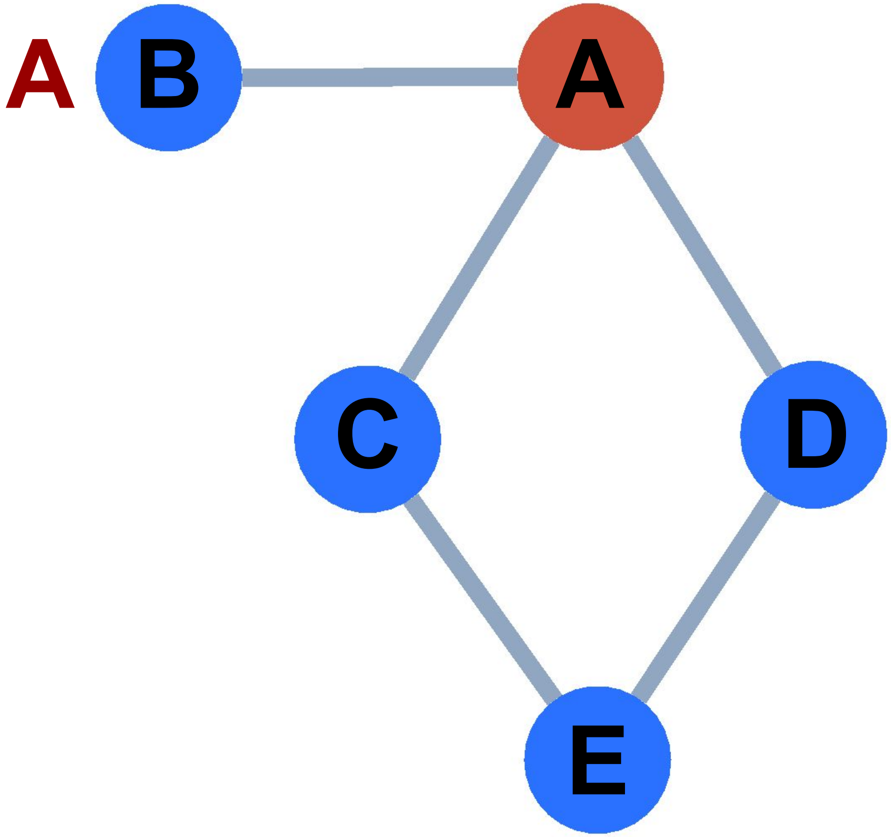


Queue

A

Plugins Developed

Node On Cycle Test



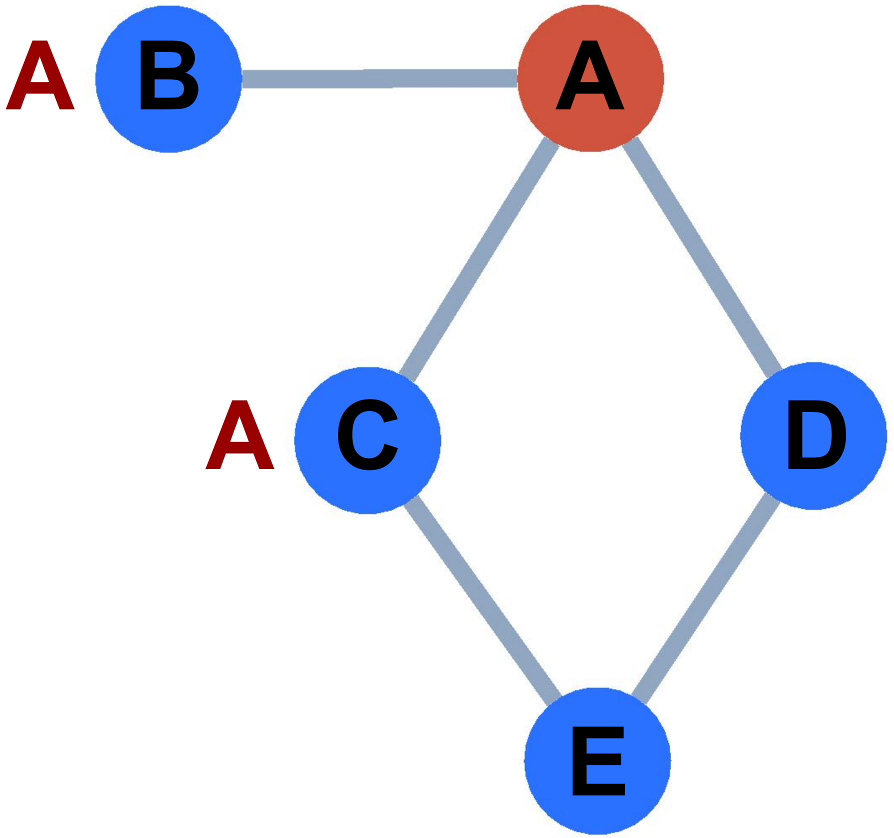
Queue

A

B

Plugins Developed

Node On Cycle Test

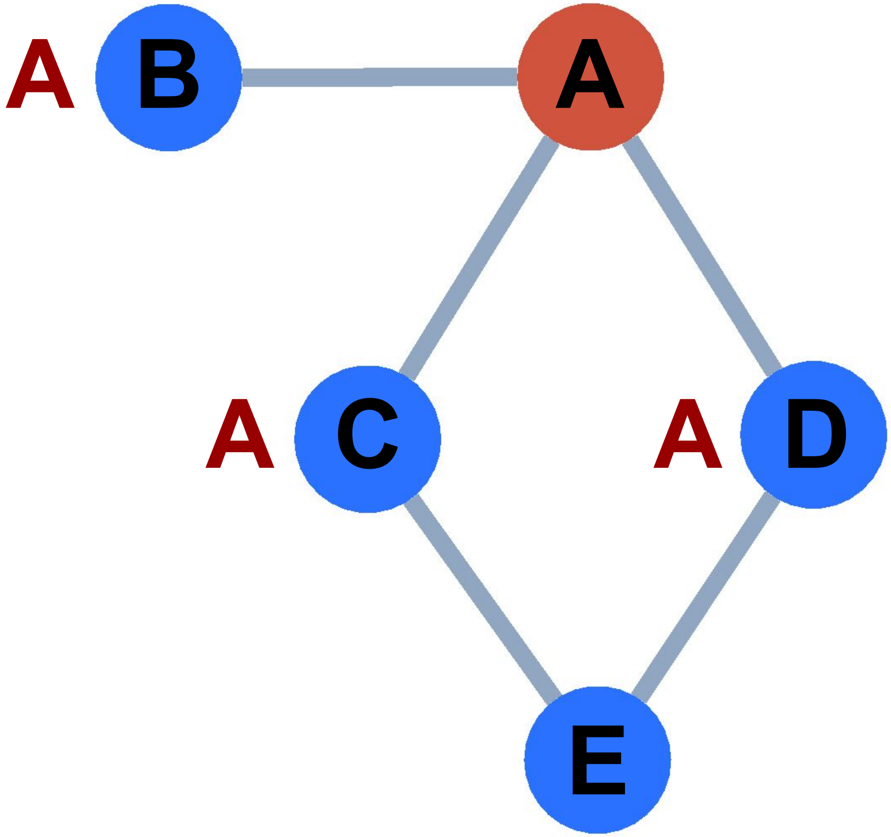


Queue

A
B
C

Plugins Developed

Node On Cycle Test

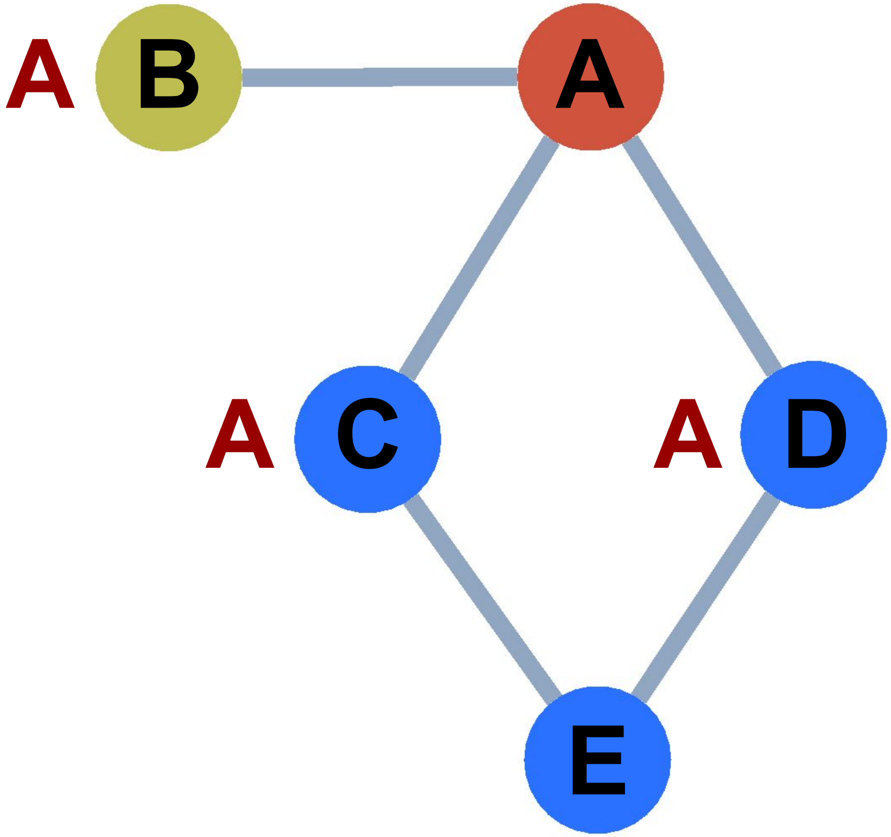


Queue

A
B
C
D

Plugins Developed

Node On Cycle Test



Queue

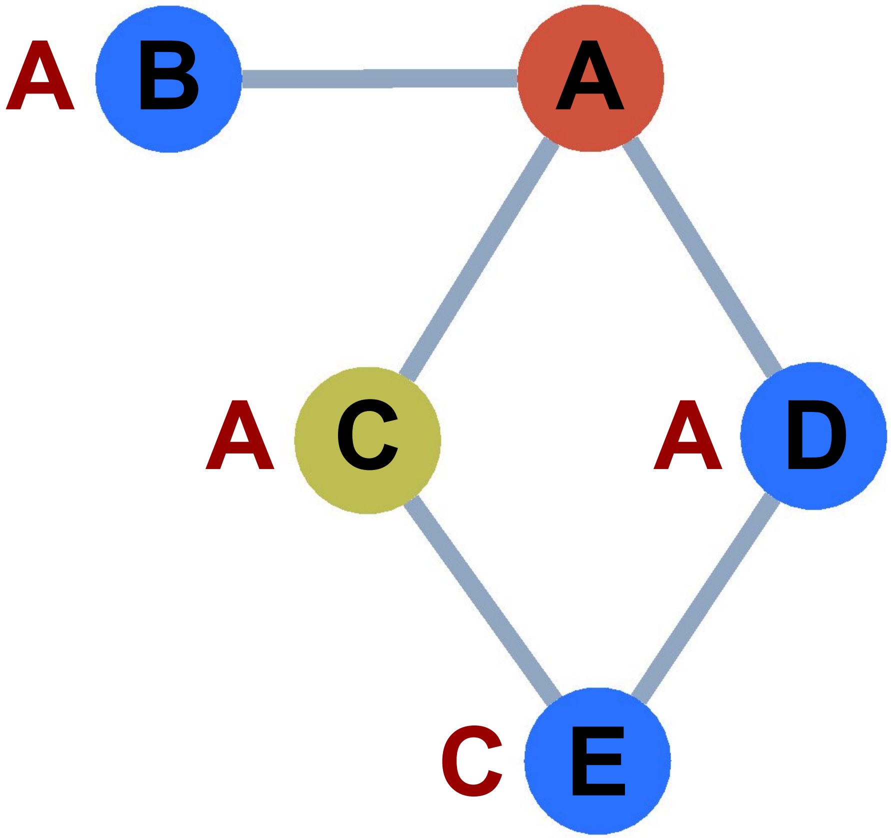
B

C

D

Plugins Developed

Node On Cycle Test



Queue

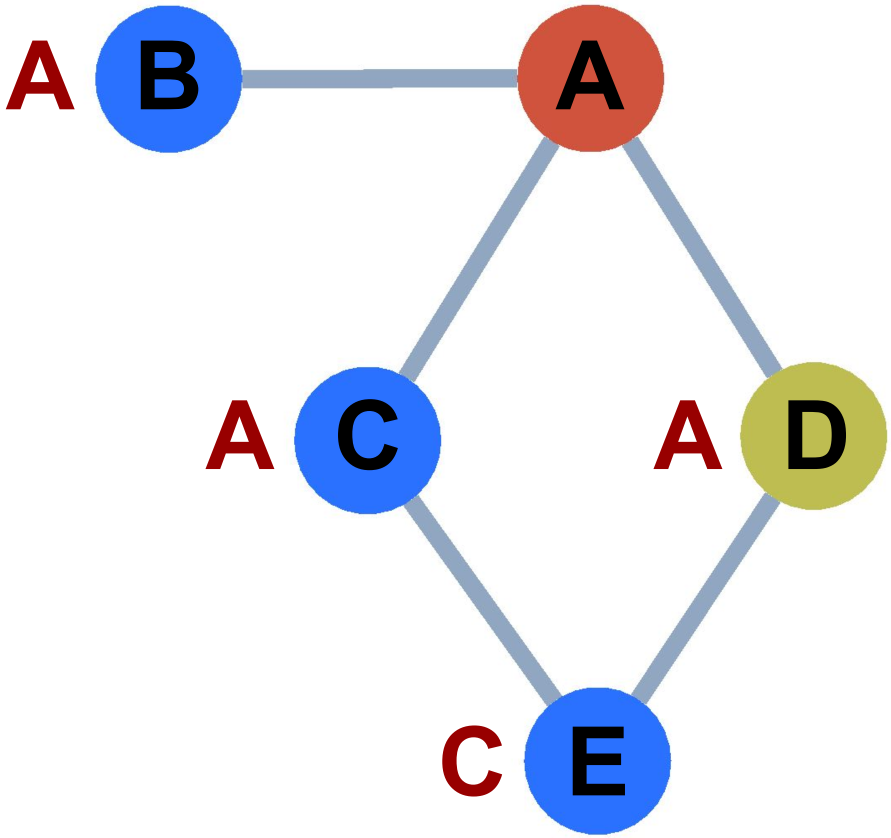
C

D

E

Plugins Developed

Node On Cycle Test



Queue

D

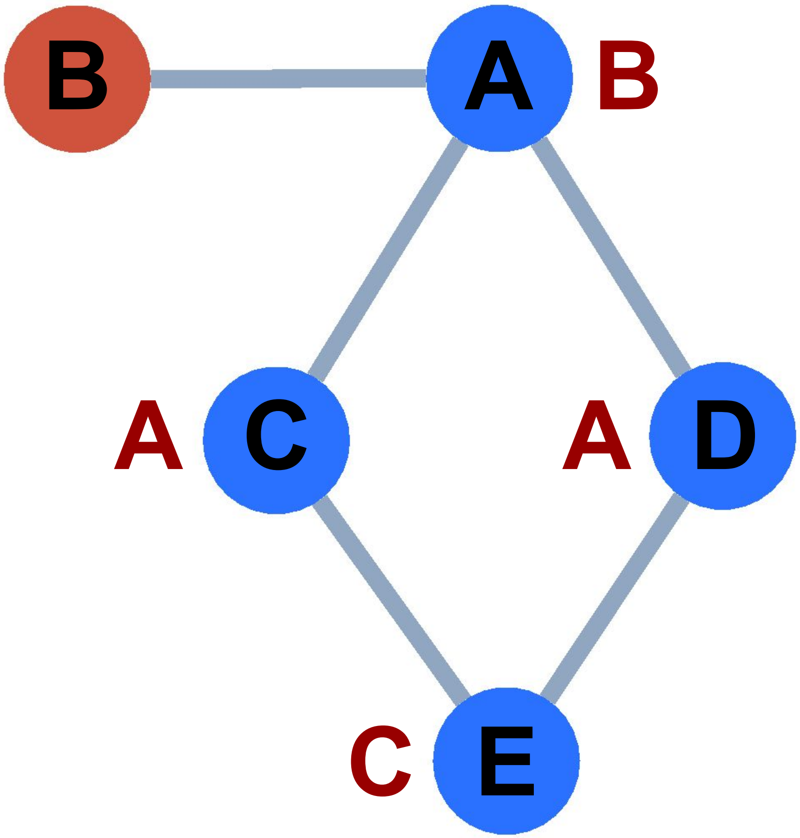
E

D-Path	**E-Path**
A	**C**
	A

Paths disjoint except for A implies A is on a cycle

Plugins Developed

Node On Cycle Test



D-Path	E-Path
A	C
B	A
	B

Paths share a node implies we found a cycle, but it doesn't include B

Image Sources

[1]https://www.top500.org/statistics/list/

[2]Auber, D., & Mary, P. (2018). Tulip (Version 5.2) [Computer software]. Bordeaux, France: LaBRI, University of Bordeaux I.

[3]https://github.com/NCAR/tulip_infiniband

[4]https://arstechnica.com/gadgets/2017/11/microsoft-and-github-team-up-to-take-git-virtual-file-system-to-macos-linux/

Slide 8

- Makefile image: http://www.iconarchive.com/show/oxygen-icons-by-oxygen-icons.org/Mimetypes-text-x-makefile-icon.html
- Dockerfile image: https://www.iconsdb.com/black-icons/text-file-5-icon.html
- README image: https://findicons.com/search/readme
- Folder image: https://dumielauxepices.net/sites/default/files/folders-clipart-computer-folder-616425-7549368.png
- Puzzle piece: http://autism-works.org/wp-content/uploads/2013/12/2012-puzzle-piece.png

Slide 26:

- Bridge: http://pngimg.com/uploads/bridge/bridge_PNG12.png