Building Scientific Workflows with Pegasus

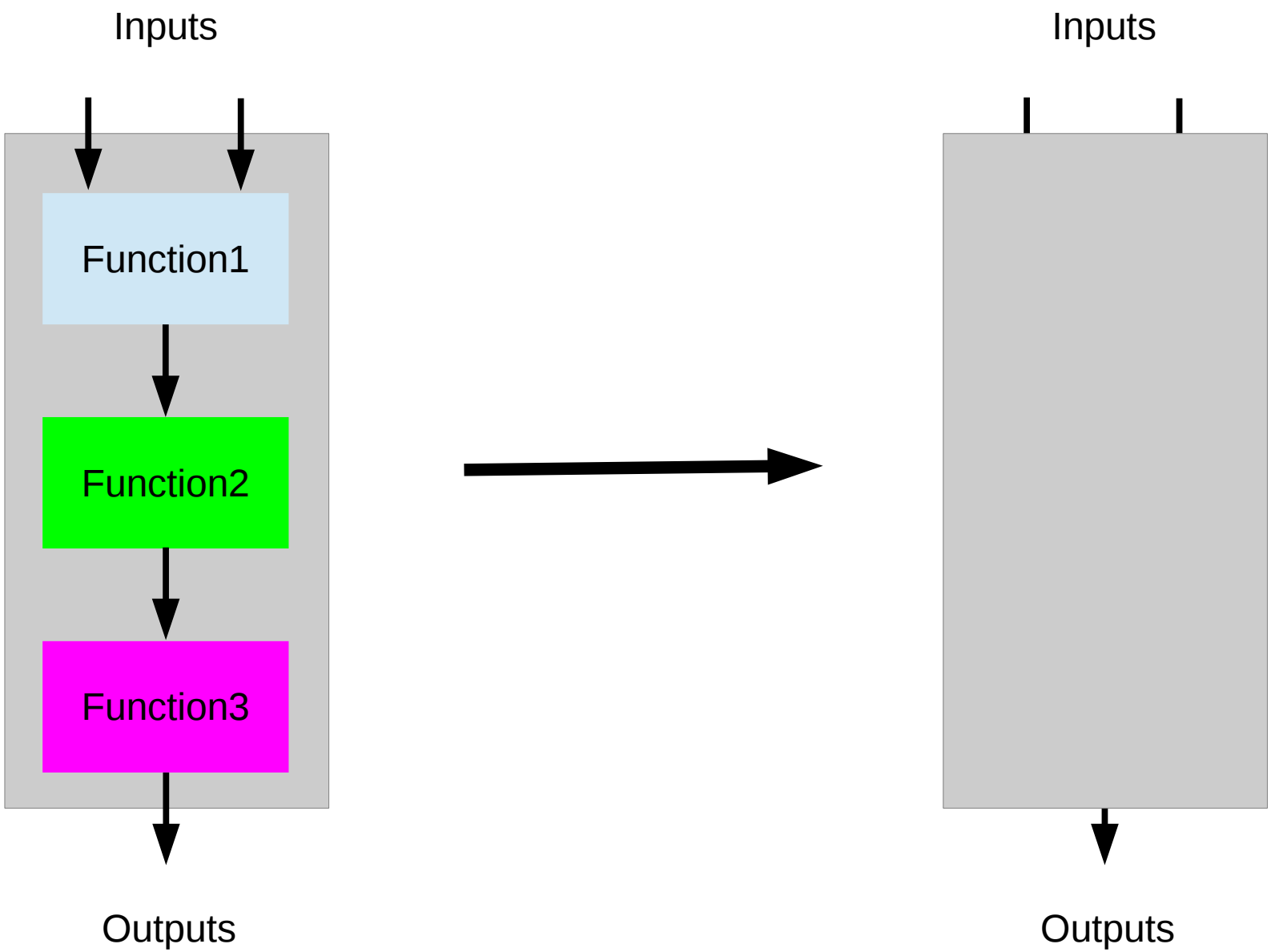
Building Blocks of Programs

B Blocks of Programs

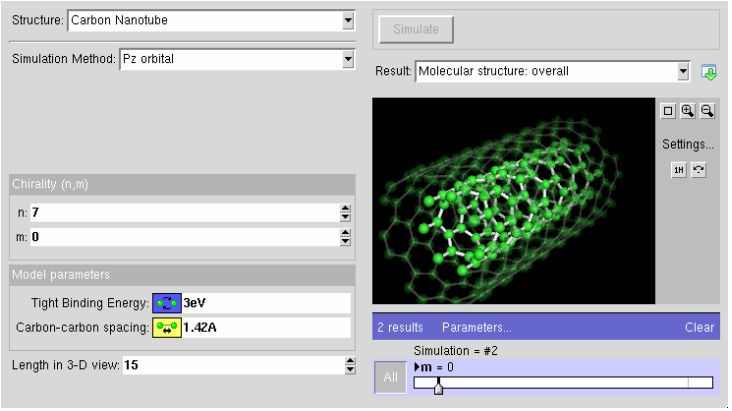
```
# goto admin interface, click on link=Support, grab ticket numbers
# if there are no tickets, set flag to create a ticket.
# give the admin user acl permission in admin interface
# goto /support/tickets grab ticket numbers
# compare to see if the user sees all tickets.
# reset user's acl permissions

proc cleanup {} {
    global manager_username

    # FIXME: delete manager and developer accounts

    # remove the user from the apps group
    #hubcheck::selenium_adminLogin \
    #    $hubcheck::optional(-hubadminuser) $hubcheck::options(-hubadminpass)
    #hubcheck::selenium_adminMemberGroups $manager_username ""
    #hubcheck::selenium_adminLogout
}

hubcheck::selenium_client_start \
    $hubcheck::options(-hubhostname):$hubcheck::options(-hubhttpsport)

hubcheck::selenium_adminLogin \
    $hubcheck::options(-hubadminuser) $hubcheck::options(-hubadminpass)

Se clickAndWait {link=Support}
Se selectAndWait {css=select#limit} {label=all}

set cssLocator {table#tktlist.adminlist > tbody > tr}
set count [Se getCSSCountN $cssLocator]
set adminticketNums ""
for {set i 1} {$i <= $count} {incr i} {
    set path "css=$cssLocator:nth-child($i) > td:nth-child(2)"

    if {[Se isElementPresent $path] == "false"} {
        # path didnt exist?
        # cleanup
        error "while retrieving ticket numbers in admin interface:\
                css location \"$path\" does not exist"
    }
    set ticketnumber [Se getText $path]
    if {![string is integer $ticketnumber] == 0} {
        # cleanup
        error "while retrieving ticket numbers in admin interface:\
                value \"$ticketnumber\" is not an integer"
    }
    lappend adminTicketNums $ticketnumber
}

set createTicket 0
if ([llength $adminTicketNums] == 0) {
    set createTicket 1
}

if ([Se isElementPresent {link=ACL}] == "false") {
    error "ACL not available"
}
Se clickAndWait {link=ACL}

set cssLocator {div#element-box > div.m > form > table.adminlist > tbody > tr}
set count [Se getCSSCountN $cssLocator]
# we read the table in reverse because the name can appear more than once.
for {set i $count} {$i > 0} {incr i -1} {
    set path "css=$cssLocator:nth-child($i) > td:nth-child(3)"

    if {[Se isElementPresent $path] == "false"} {
        # path didnt exist?
        # cleanup
        error "while retrieving acl users in admin interface:\
                css location \"$path\" does not exist"
    }
    set user ""
    regexp {(\w+) \(\d+\)} [Se getText $path] match user

    if {$user == $hubcheck::options(-hubadminuser)} {
        break
    }
}

set deleteperms 0

# adjust permissions
if ($i == 0) {
    # add a new entry
    Se type {css=input#aro_alias} $hubcheck::options(-hubadminuser)
    set path {css=div#element-box > div.m > form > table.adminlist > tfoot > tr}
    Se click "$path > td:nth-child(6) > input"
    Se clickAndWait "$path > td:nth-child(12) > input"
    set deleteperms 1
} else {
    # found the user in the list, store their permissions
    foreach col {list 5 6 7 8 9 10 11} {
        set path "css=$cssLocator:nth-child($i) > td:nth-child($col)"
        set perms($col) [Se isElementPresent "$path > a > img\[@alt=\"Yes\"\]"]
    }

    # ensure they have ticket read permission (column 5)
    if ($perms(5) == "false") {
        set path "css=$cssLocator:nth-child($i) > td:nth-child(5) > a"
        Se clickAndWait $path
        # check to make sure click worked
        set path "$path > img\[@alt=\"Yes\"\]"
        if ([Se isElementPresent $path] == "false") {
            cleanup
            error "while updating acl permission: clicking $path failed"
        }
    }
    set deleteperms 0
}

hubcheck::selenium_adminLogout

# get tickets from the web site.
hubcheck::selenium_siteLogin \
    $hubcheck::options(-hubadminuser) $hubcheck::options(-hubadminpass)

Se open /support/tickets
Se selectAndWait {css=select#limit} {label=all}

set cssLocator {table#tktlist > tbody > tr}
set count [Se getCSSCountN $cssLocator]
set webTicketNums ""
for {set i 1} {$i <= $count} {incr i} {
    set path "css=$cssLocator:nth-child($i) > td:nth-child(1)"

    if {[Se isElementPresent $path] == "false"} {
        # path didnt exist?
        cleanup
        error "while retrieving ticket numbers in admin interface:\
                css location \"$path\" does not exist"
    }
    set ticketnumber [Se getText $path]
    if {![string is integer $ticketnumber] == 0} {
        # cleanup
        error "while retrieving ticket numbers in admin interface:\
                value \"$ticketnumber\" is not an integer"
    }
    lappend webTicketNums $ticketnumber
}

# are the ticket numbers the same?
set result ""
if ([llength $adminTicketNums] == [llength $webTicketNums]) {
    foreach aT $adminTicketNums {
        if ([lsearch -exact $webTicketNums $aT] == -1) {
            lappend result $aT
        }
    }
}

hubcheck::selenium_siteLogout

# revert user's permissions if necessary

if ($perms(5) == "false") || ($deleteperms == 1)) {
    # FIXME: this looks incomplete
}

hubcheck::selenium_adminLogin \
    $hubcheck::options(-hubadminuser) $hubcheck::options(-hubadminpass)

Se clickAndWait {link=Support}
Se clickAndWait {link=ACL}

set path ""
set cssLocator {div#element-box > div.m > form > table.adminlist > tbody > tr}
set count [Se getCSSCountN $cssLocator]
# we read the table in reverse because the name can appear more than once.
for {set i $count} {$i > 0} {incr i -1} {
    set path "css=$cssLocator:nth-child($i) > td:nth-child(3)"

    if {[Se isElementPresent $path] == "false"} {
        # path didnt exist?
        # cleanup
        error "while retrieving acl users in admin interface:\
                css location \"$path\" does not exist"
    }
    set user ""
    regexp {(\w+) \(\d+\)} [Se getText $path] match user

    if {$user == $hubcheck::options(-hubadminuser)} {
        break
    }
}

# adjust permissions
if ($i != 0) {
    if ($deleteperms == 1) {
        # delete the entry
        Se click "$path > td:nth-child(1) > input"
        Se clickAndWait "css=td#toolbar-delete > a"
    } else {
        # restore their permissions for column 5,
        # since that is the only one we changed
        set path "css=$cssLocator:nth-child($i) > td:nth-child(5) > a"
        Se clickAndWait $path

        # ensure the permission was restored
        # check to make sure click worked
        set path "$path > img\[@alt=\"No\"\]"
        if ([Se isElementPresent $path] == "false") {
            cleanup
            error "while updating acl permission in cleanup: clicking $path failed"
        }
    }
}

hubcheck::selenium_adminLogout

#cleanup
hubcheck::selenium_client_stop

if ([llength $result] != 0) {
    return "The following tickets were missing from the web interface: $result"
} else {
    return ""
}
```

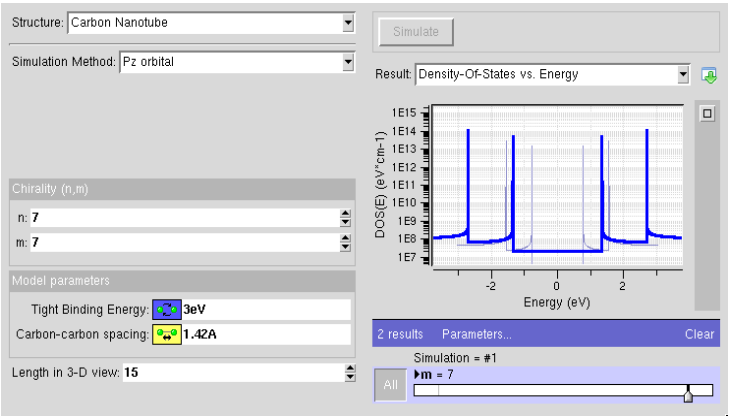
```
import unittest
import sys

from hubcheck.webdriver import Webdriver
from hubcheck.pageobjects import SupportNeedHelp

class website_needhelp_link_exists(unittest.TestCase):

    def setUp(self):

        locators    = "@@tcl:HCVAR options(-locators)@@"
        hubname     = "@@tcl:HCVAR options(-hubhostname)@@"
        hubport     = "@@tcl:HCVAR options(-hubhttpsport)@@"
        seHost      = "@@tcl:HCVAR options(-seleniumrc_host)@@"
        sePort      = "@@tcl:HCVAR options(-seleniumrc_port)@@"
        profilesdir = "@@tcl:HCVAR options(-profilesdir)@@"

        url = "https://%s:%s" % (hubname,hubport)

        self.wd = Webdriver(seHost,sePort,profilesdir)
        self.wd.locatorKey = locators
        self.wd.connect(url)

    def testSupportNeedHelpLinkExists(self):
        # click the need help link,
        # to see if the widget exists
        po = SupportNeedHelp()
        po.open()
        po.close()

    def tearDown(self):
        self.wd.close()
```

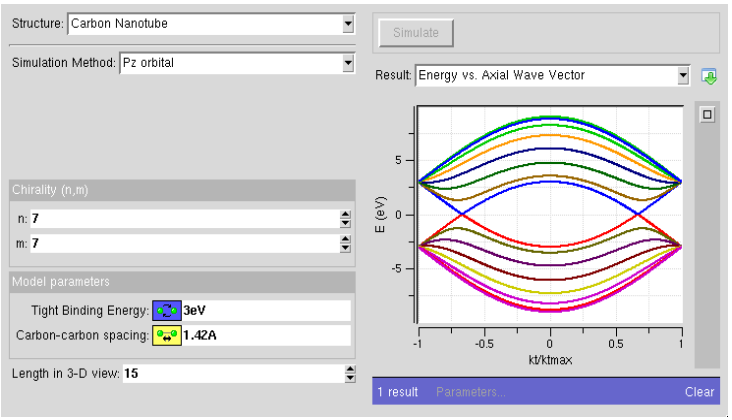
Building Blocks of Programs



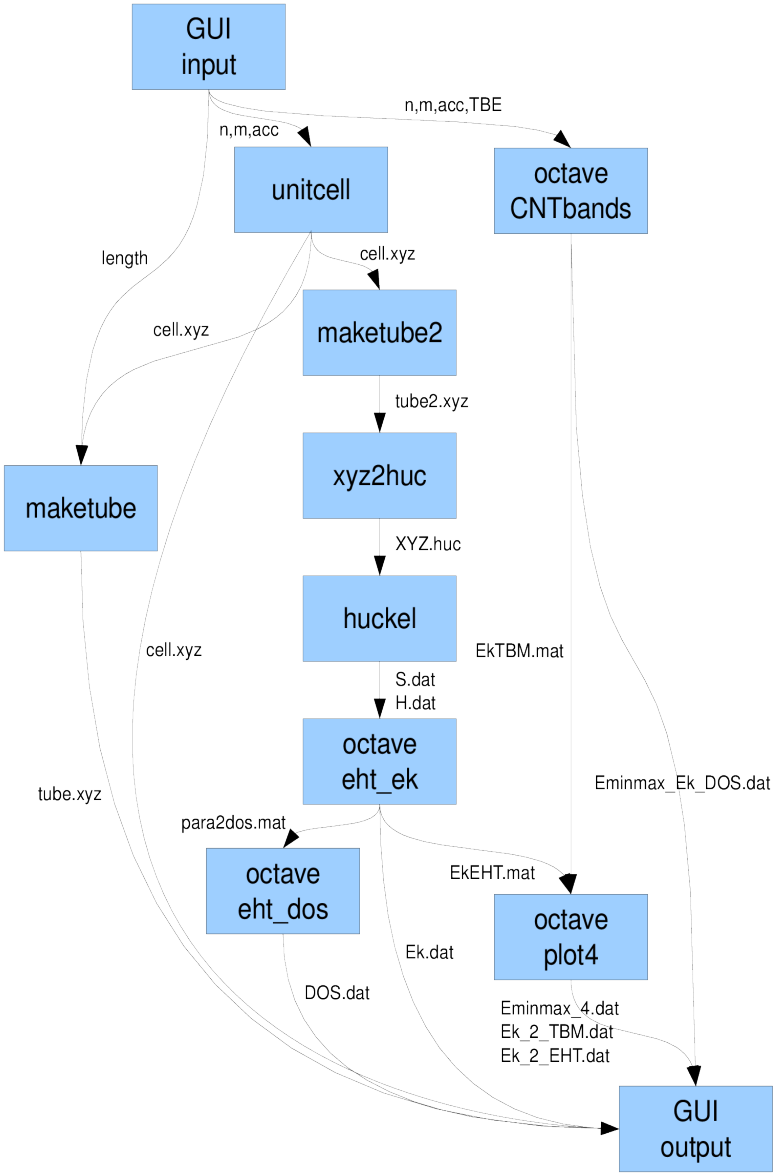
Building Blocks of Science



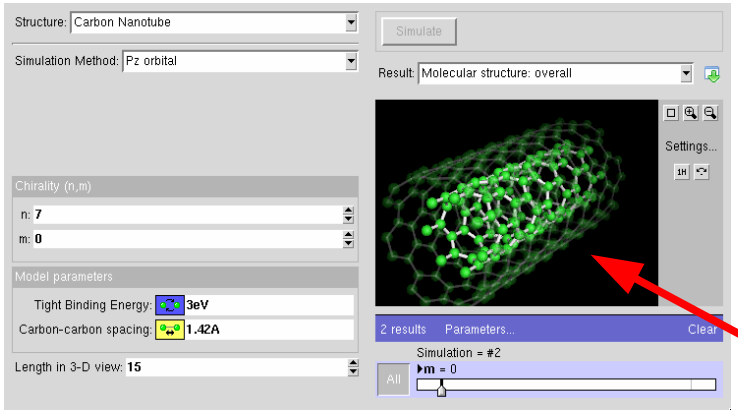


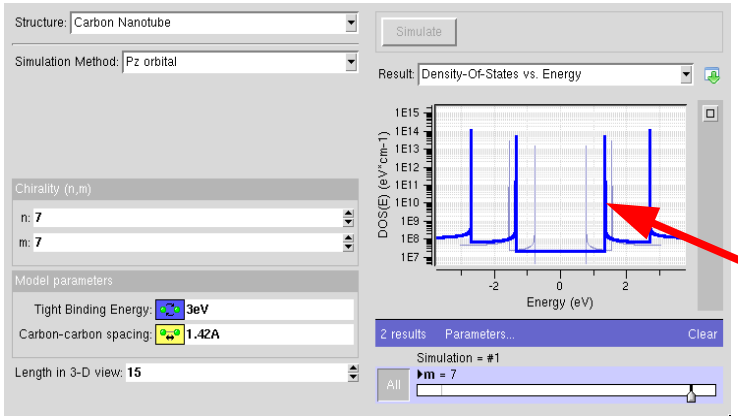


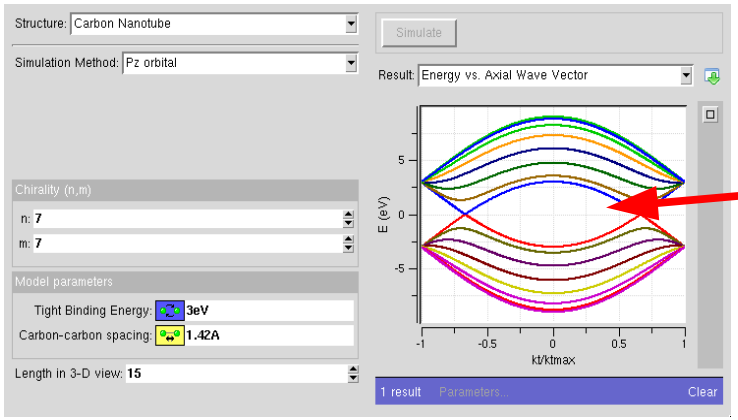
Building Blocks of Science

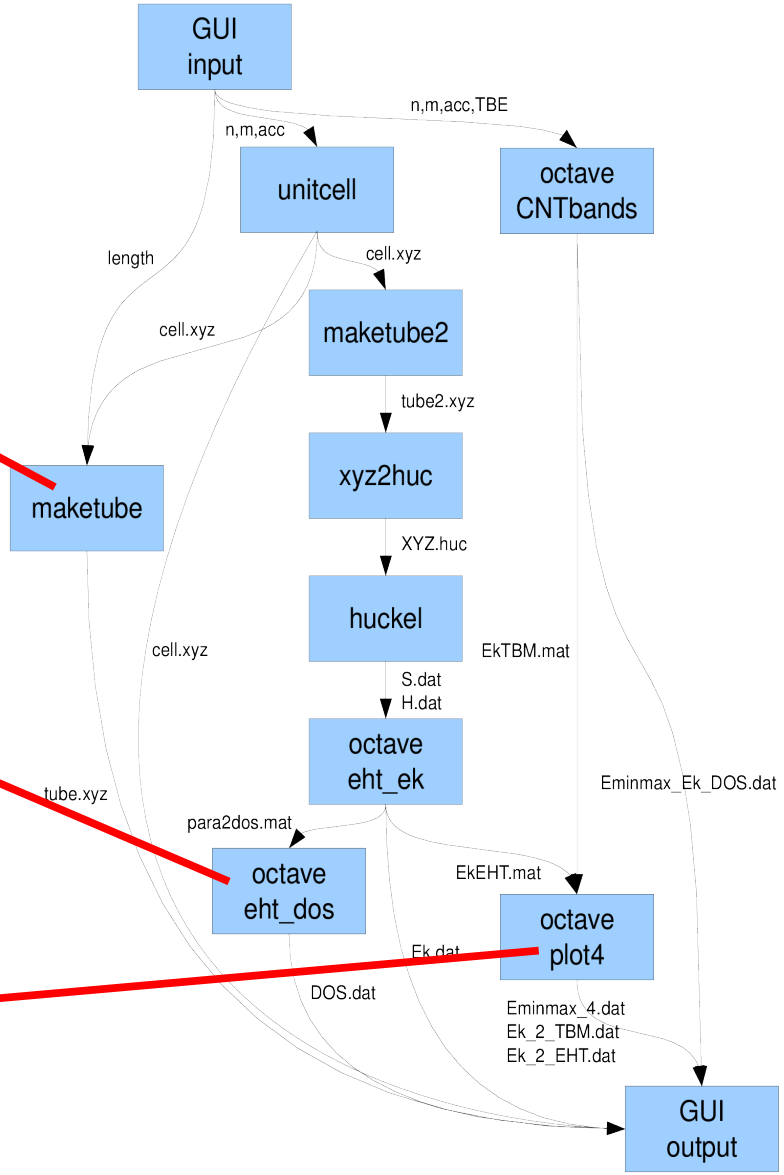


Building Blocks of Science





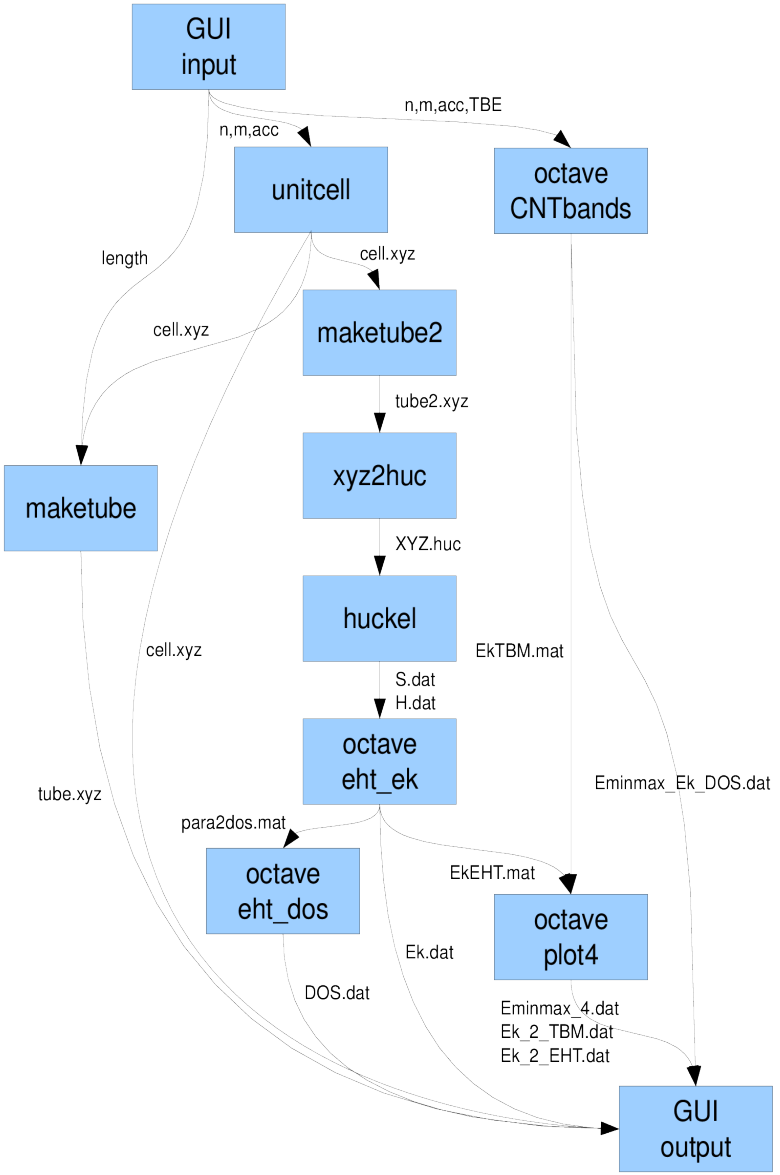




Types of Workflows

Sequential Workflows

- Execute steps in order until all of the work has been completed
- Could include activities that run in parallel



CNTBands

Science Domain: Nanoelectronics

Scientists: Lundstrom et al. (Purdue)

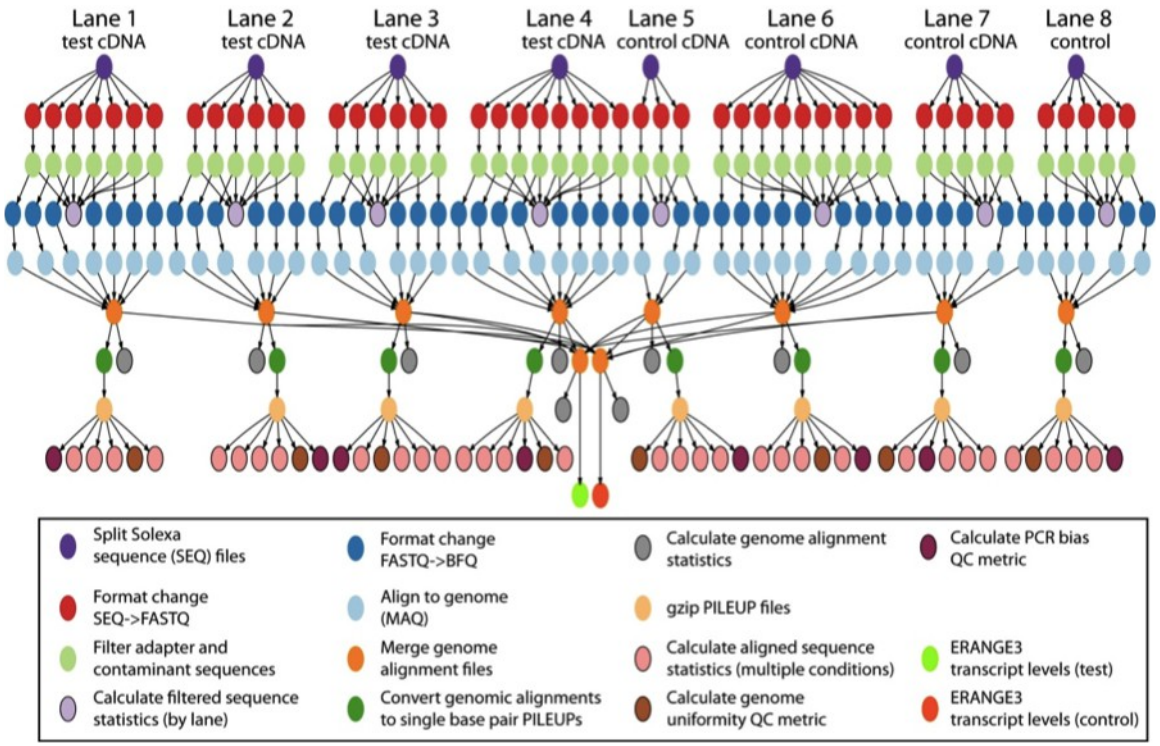
https://nanohub.org/resources/cntbands-ext

Types of Workflows

Wideband Workflows

- Execute the same function many (1000's) of times

- Massively parallel
- Scatter / Gather
- Sweeps

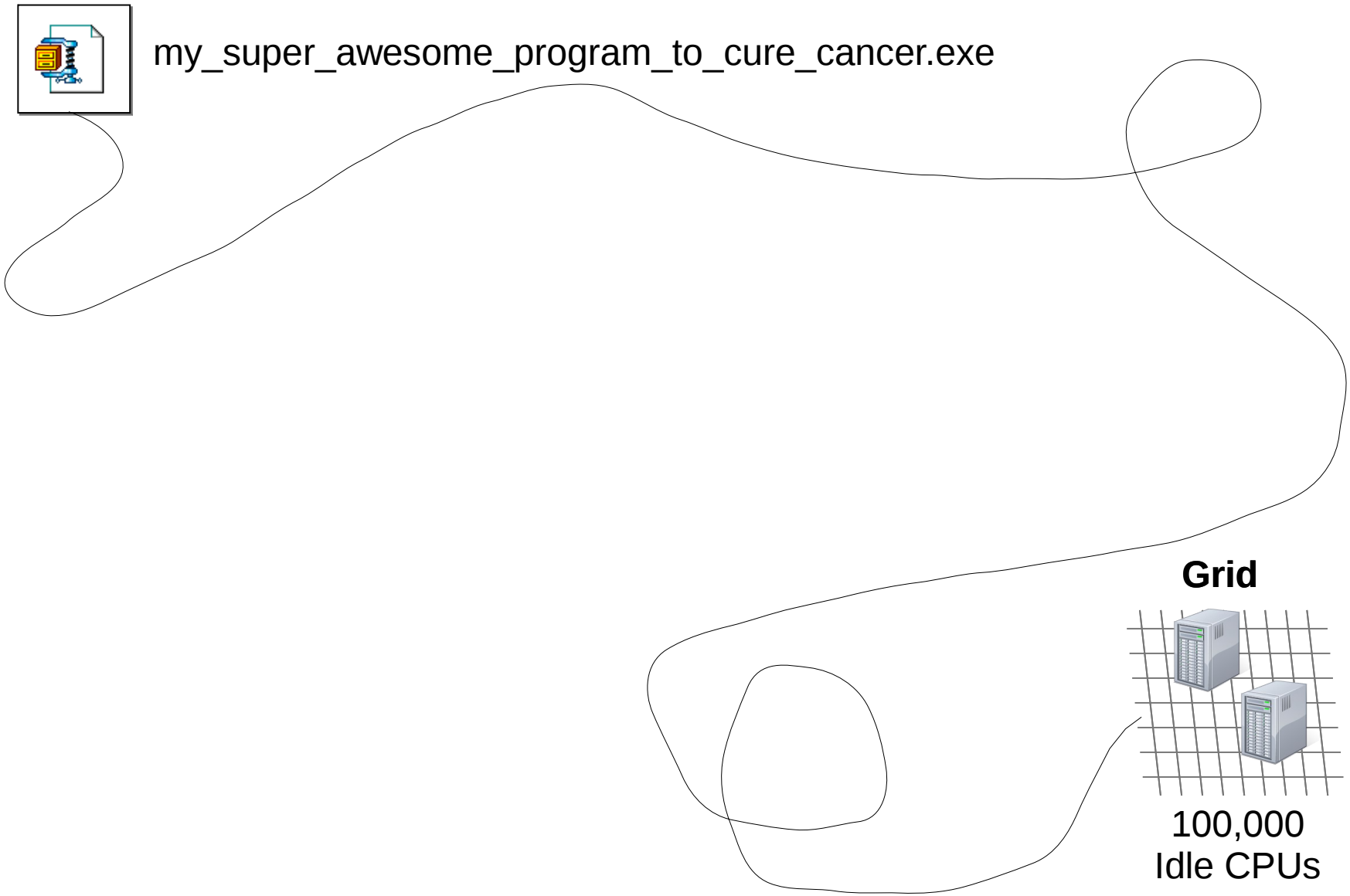


Epigenomics

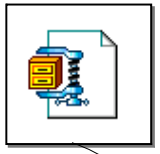
Science Domain: Bioinformatics

Scientists: Ben Berman et al. (USC)

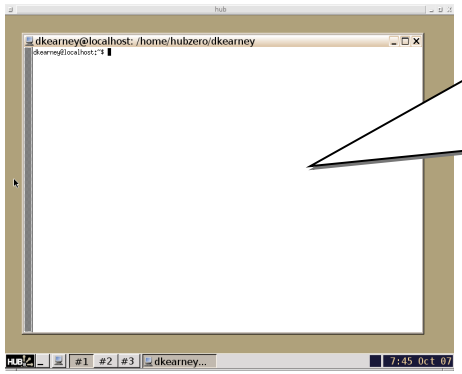
Big Work, Big Computers



Big Work, Big Computers

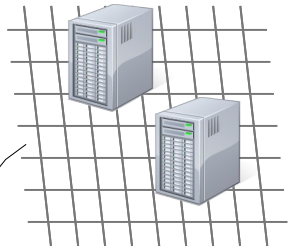


my_super_awesome_program_to_cure_cancer.exe



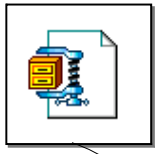

```
$ ssh me@the-grid.com
me@the-grid.com's password:

password rejected
Error #9: Your not cool enough
```

Grid

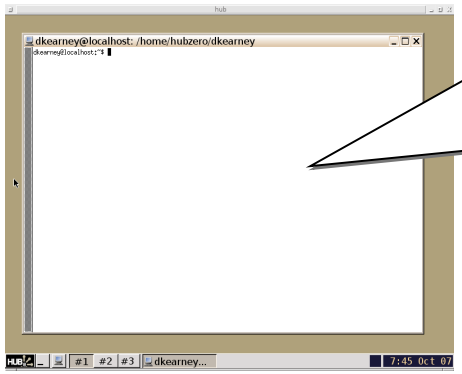


100,000
Idle CPUs

Big Work, Big Computers



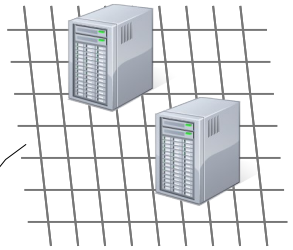
my_super_awesome_program_to_cure_cancer.exe



$ ssh me@the-grid.com
me@the-grid.com's password:

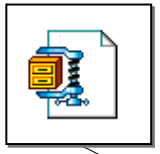
$ make all install
Library Version Errors...

Grid



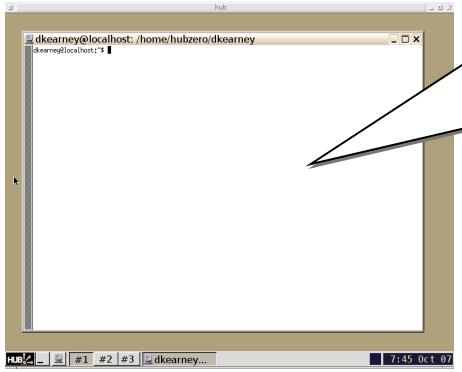
100,000
Idle CPUs

Big Work, Big Computers

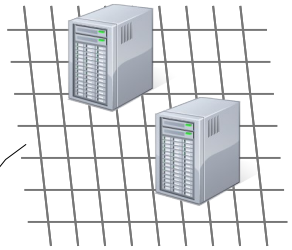


my_super_awesome_program_to_cure_cancer.exe

```
$ ssh me@the-grid.com
me@the-grid.com's password:

$ ./my_super_awesome_program_to_cure_cancer.exe \
    datafile1 datafile2
```

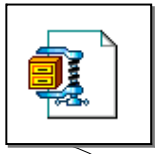


Grid

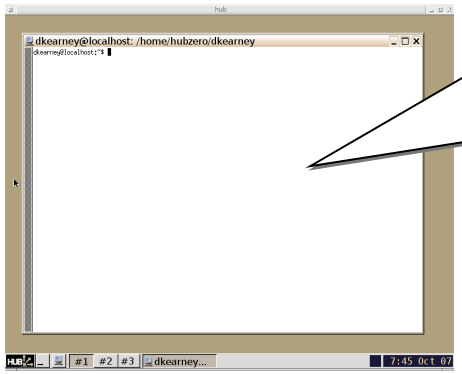


100,000
Idle CPUs

Big Work, Big Computers



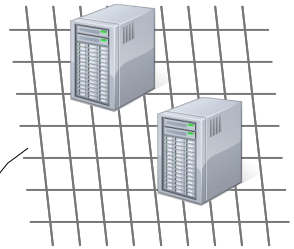
my_super_awesome_program_to_cure_cancer.exe



$ ssh myfriend@the-grid.com
myfriend@the-grid.com's password:

password rejected
Error #9: Your friend isn't cool enough either!

Grid

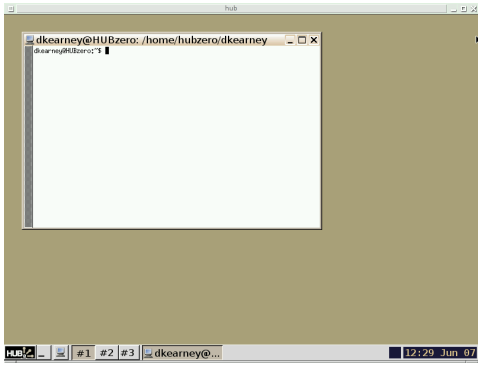


100,000
Idle CPUs

Project Files

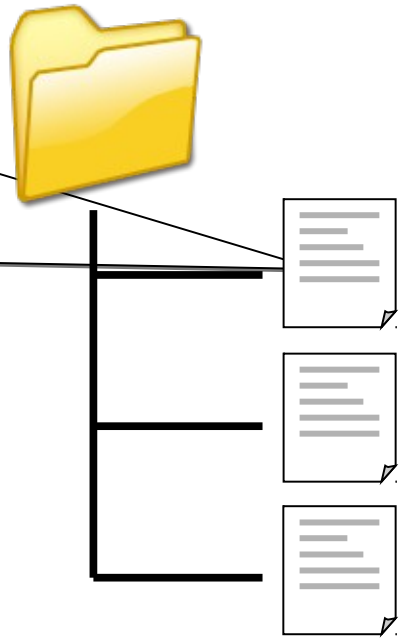
User's Workspace Terminal

HUBzero Infrastructure



Tool Session Containers

```
$ cat /apps/pegtut/current/bin/sayhi.sh

#!/bin/bash

# output something on stdout
echo "Hello `cat ${1}`!"

# print greeting to a file
echo "Hello `cat ${1}`!" >f.b
```



sayhi.sh

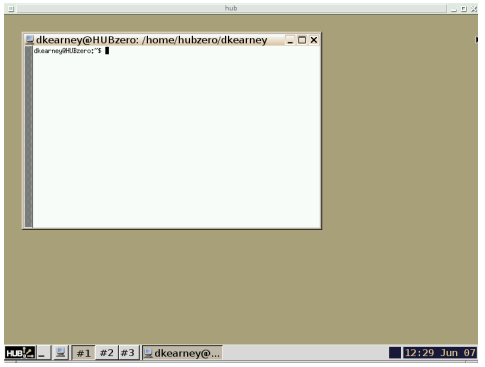
inquire.sh

f.a

Project Files

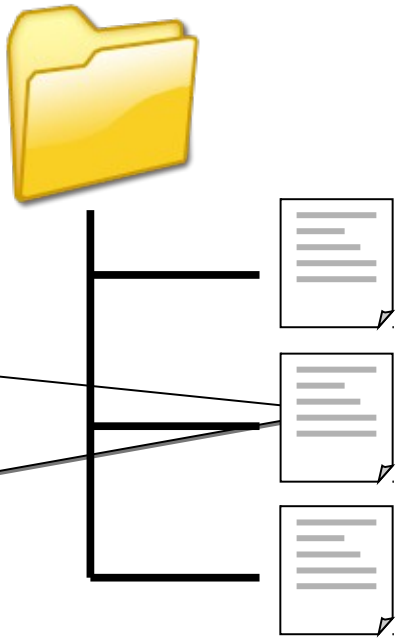
User's Workspace Terminal

HUBzero Infrastructure



Tool Session Containers

```
$ cat /apps/pegtut/current/bin/inquire.sh

#!/bin/bash

# output some thing to stdout
echo "`cat ${1}` How are you?"

# print greeting to a file
echo "`cat ${1}` How are you?" >f.c
```



sayhi.sh

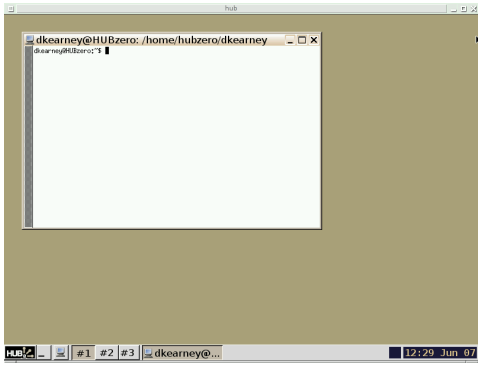
inquire.sh

f.a

Project Files

User's Workspace Terminal

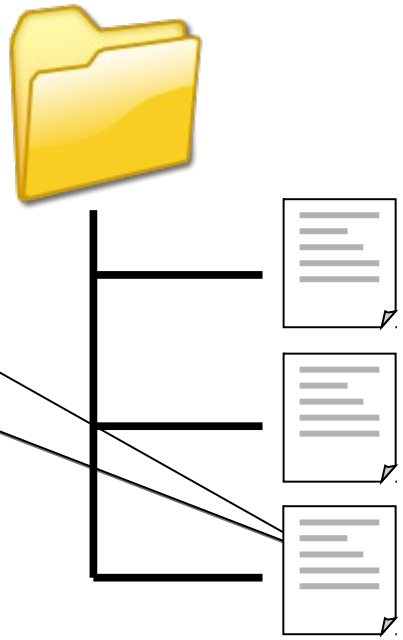
HUBzero Infrastructure



Tool Session Containers

$ cat f.a

pete



sayhi.sh

inquire.sh

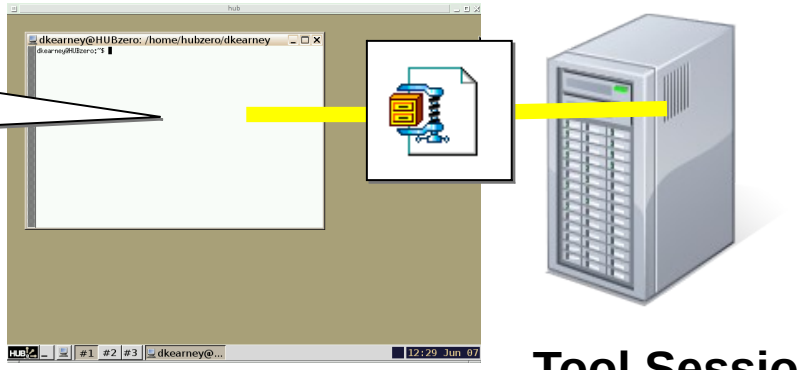
f.a

Running locally in a Workspace

User's Workspace Terminal

$ /apps/pegtut/current/bin/sayhi.sh f.a

HUBzero Infrastructure



Tool Session Containers

Submit Proxy

Grid

Grid

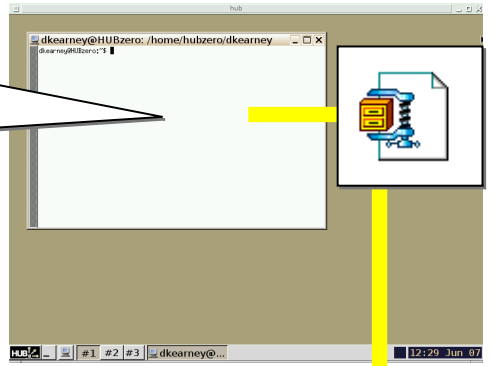
Grid

Submitting from a Workspace

User's Workspace Terminal

$ submit /apps/pegtut/current/bin/sayhi.sh f.a

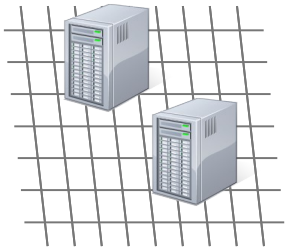
HUBzero Infrastructure



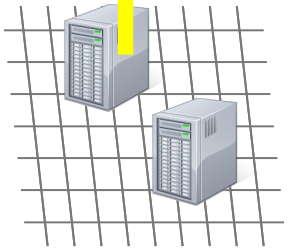
Tool Session Containers

Submit Proxy

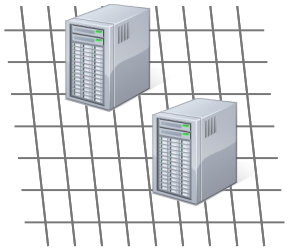
Grid



Grid



Grid

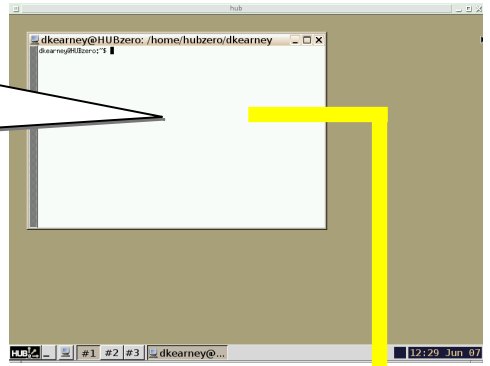


Submitting from a Workspace

User's Workspace Terminal

$ submit /apps/pegtut/current/bin/sayhi.sh f.a

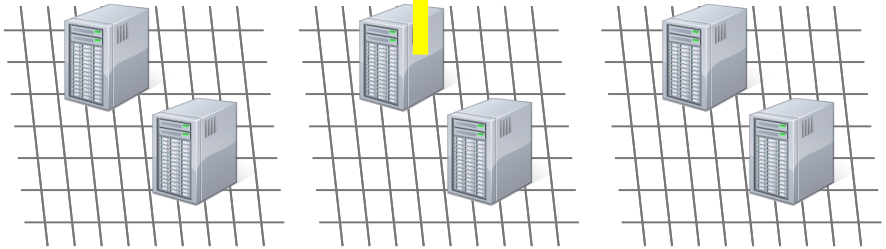
HUBzero Infrastructure



Tool Session Containers

Submit Proxy

Grid **Grid** **Grid**

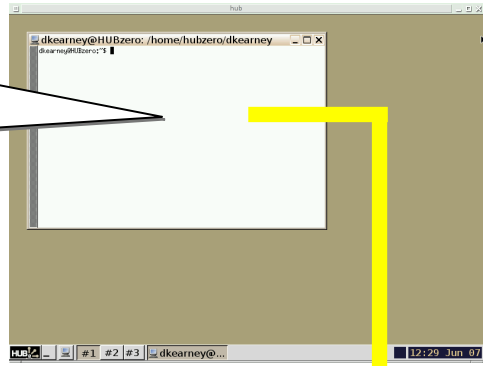


Submitting from a Workspace

User's Workspace Terminal

$ submit /apps/pegtut/current/bin/sayhi.sh f.a

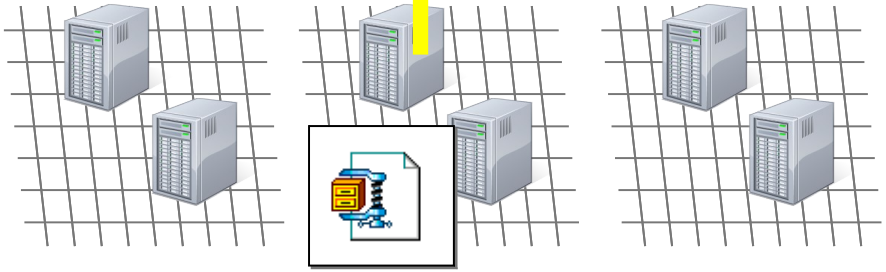
HUBzero Infrastructure



Tool Session Containers

Submit Proxy

Grid **Grid** **Grid**

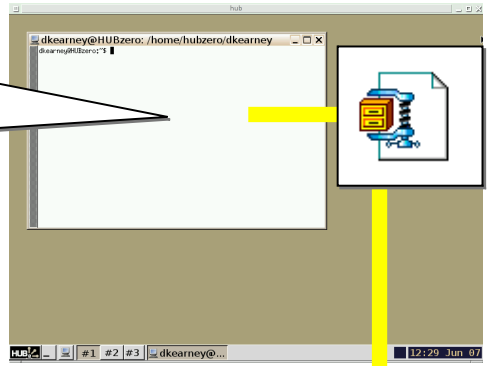


Submitting from a Workspace

User's Workspace Terminal

$ submit /apps/pegtut/current/bin/sayhi.sh f.a

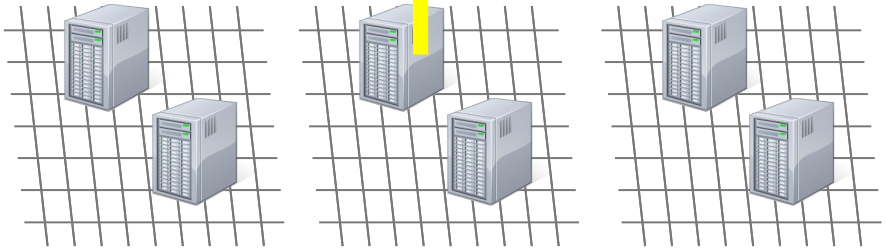
HUBzero Infrastructure



Tool Session Containers

Submit Proxy

Grid **Grid** **Grid**

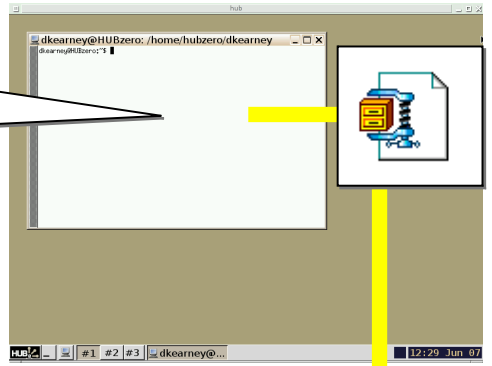


Submitting from a Workspace

User's Workspace Terminal

```
$ submit --local \
    /apps/pegtut/current/bin/sayhi.sh f.a
```

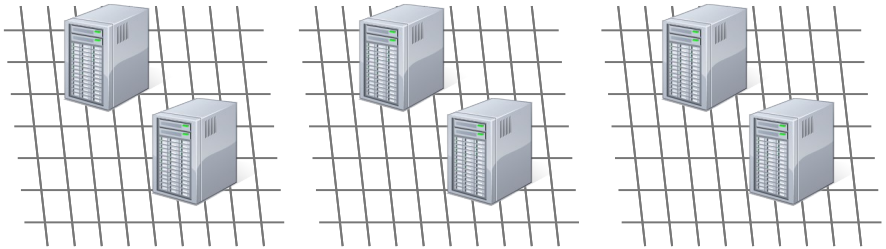
HUBzero Infrastructure



Tool Session Containers

Submit Proxy

Grid **Grid** **Grid**

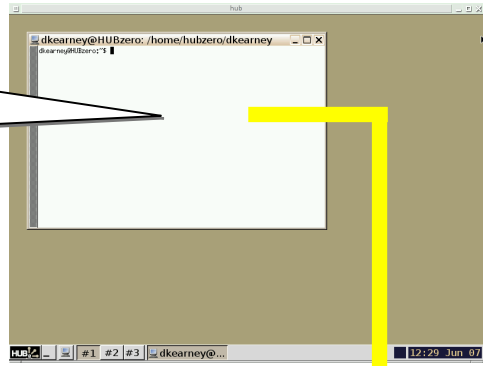


Submitting from a Workspace

User's Workspace Terminal

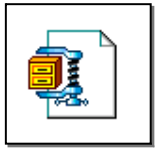
```
$ submit --local \
    /apps/pegtut/current/bin/sayhi.sh f.a
```

HUBzero Infrastructure

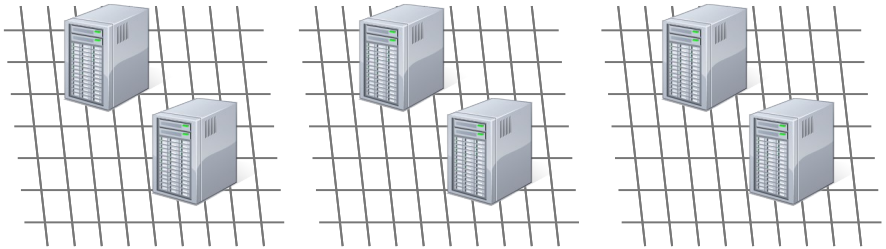


Tool Session Containers

Submit Proxy



Grid **Grid** **Grid**

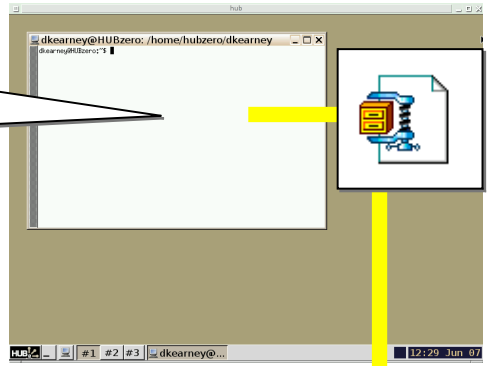


Submitting from a Workspace

User's Workspace Terminal

```
$ submit --local \
    /apps/pegtut/current/bin/sayhi.sh f.a
```

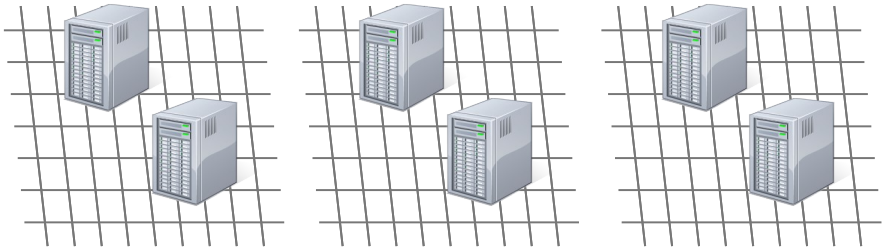
HUBzero Infrastructure



Tool Session Containers

Submit Proxy

Grid **Grid** **Grid**

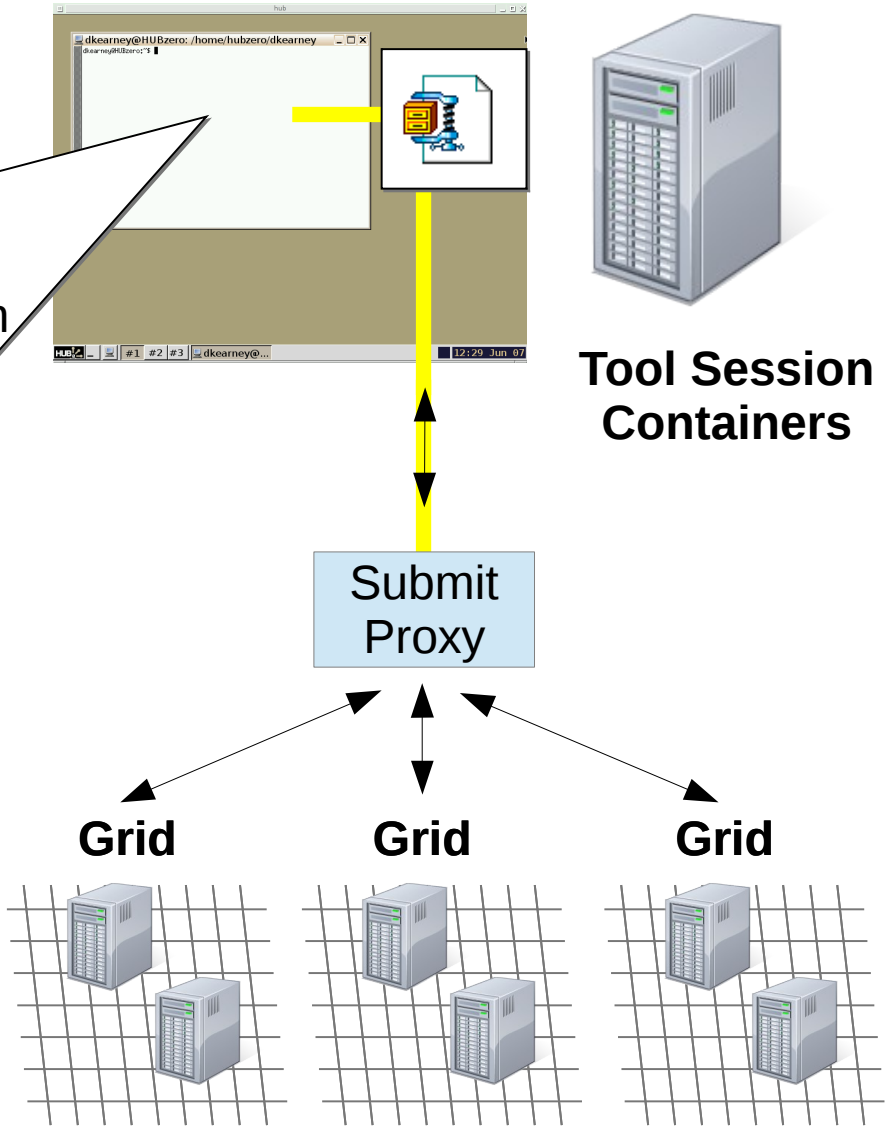


Submitting from a Workspace

User's Workspace Terminal

```
$ submit --help

  -l, --local          Execute command
                        locally
  -v, --venue          Remote job destination
  -i, --inputfile      Input file
  -p, --parameters     Parameter sweep
                        variables.
  -d, --data           Parametric variable
                       data - csv format

  ...
```

HUBzero Infrastructure



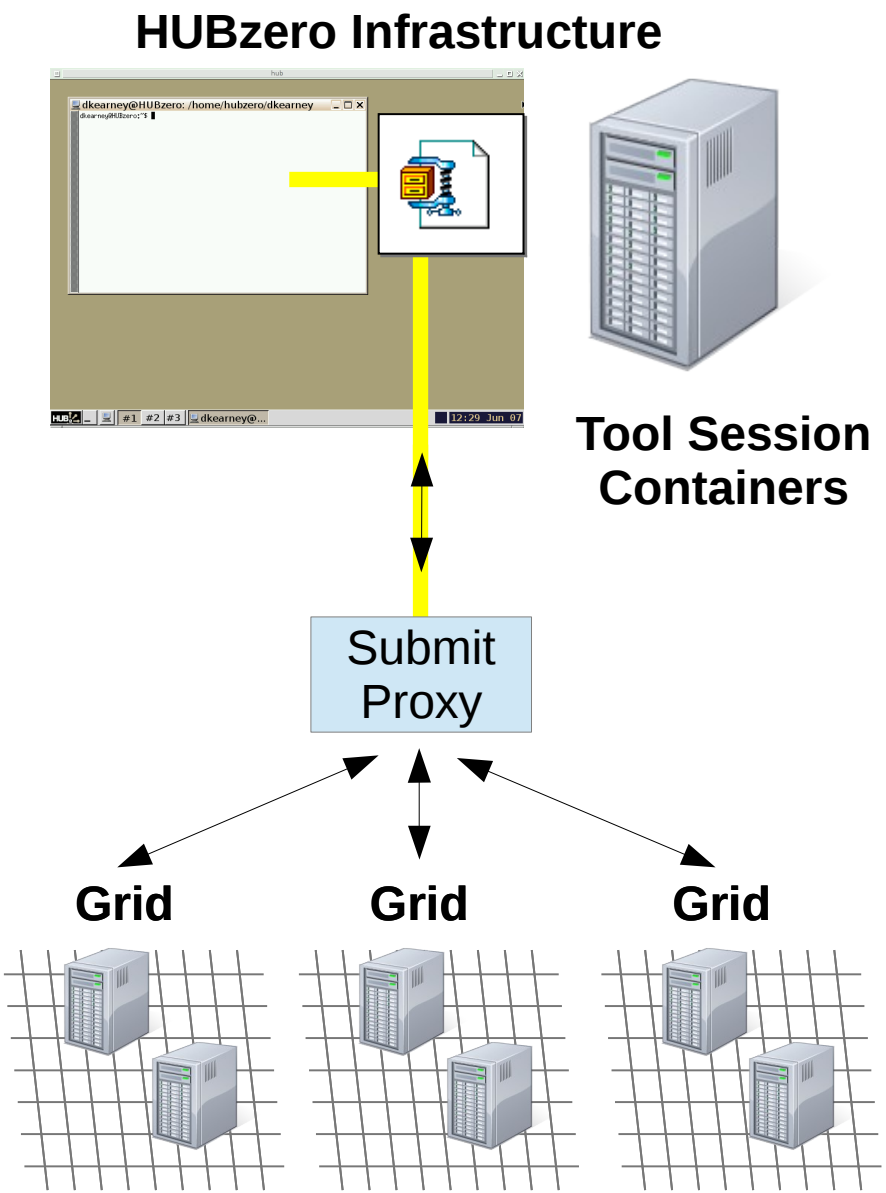
What commands can I submit?

Submitting locally with --local

- Any executable can be used when submitting locally

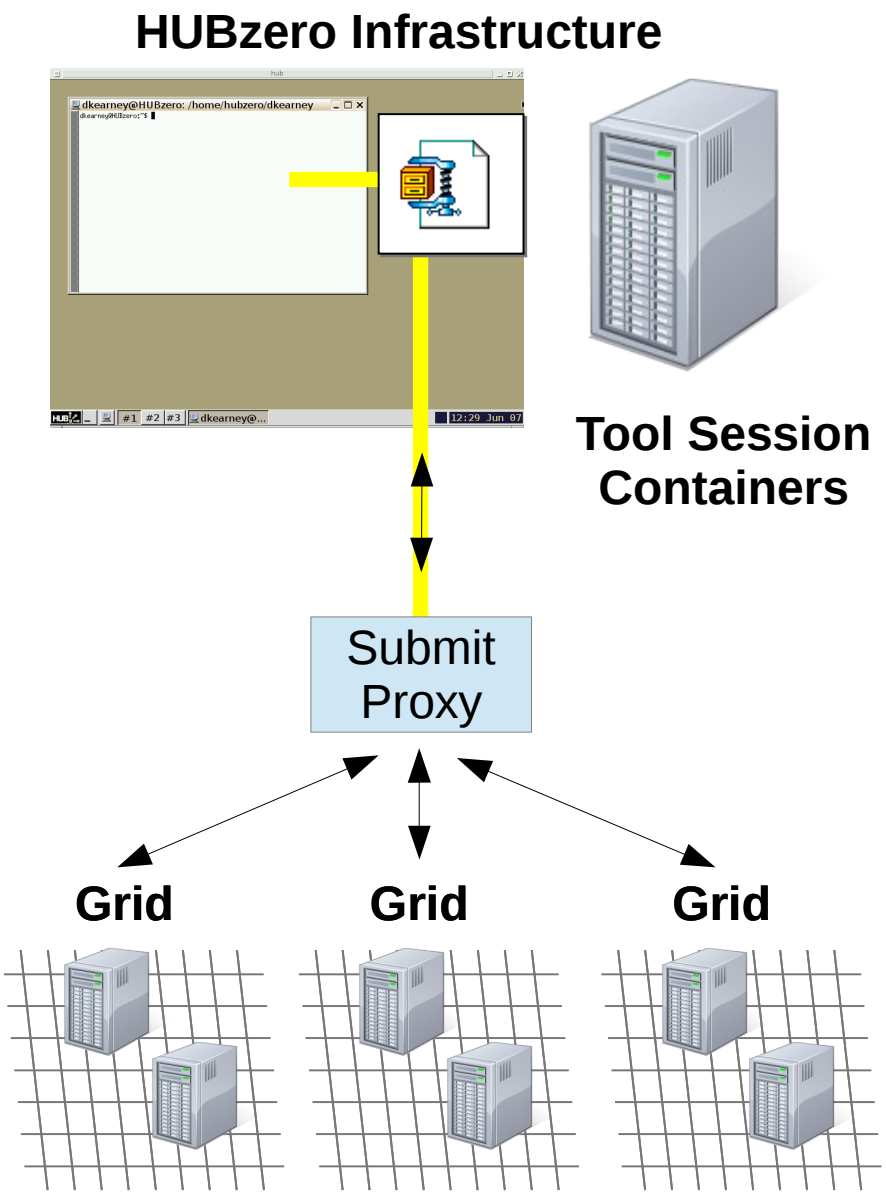
Submitting to the grid

- Only executables staged in /apps can be submitted to the grid



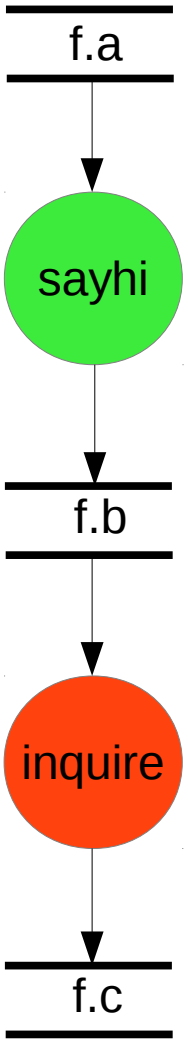
Try submitting a command

```
$ submit --local \
    /apps/pegtut/current/bin/sayhi f.a

Hello pete!
```

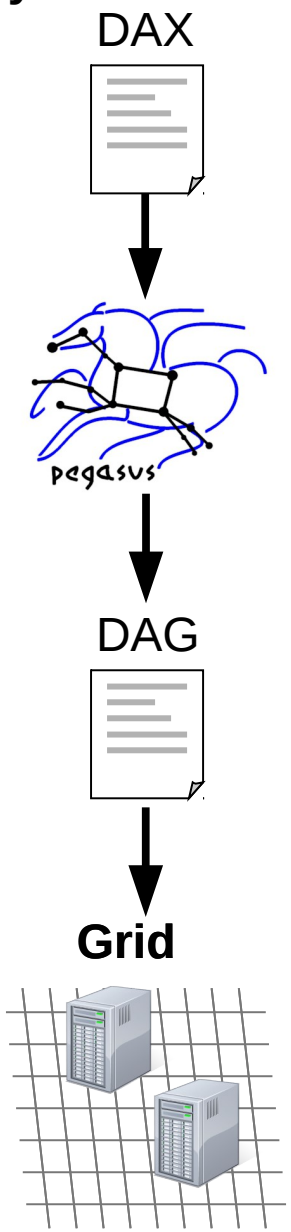


How does Pegasus Work?

If you can draw it ...

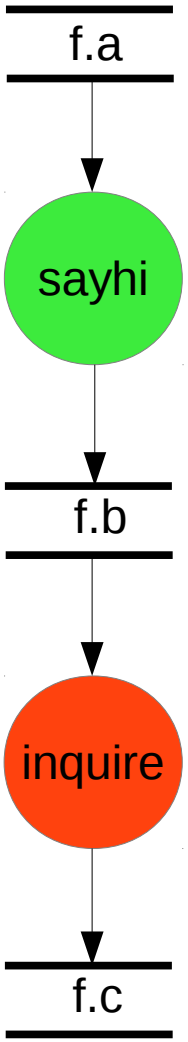


… they can make it run



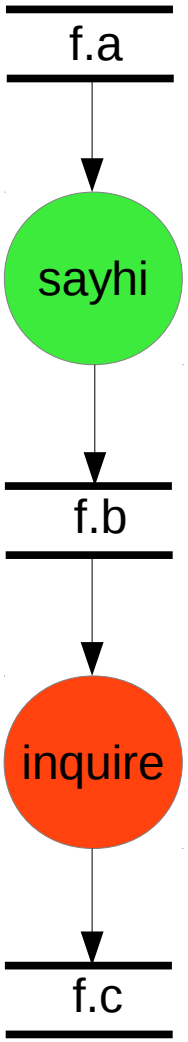
How does Pegasus Work?

Step 1. Draw the workflow



How does Pegasus Work?

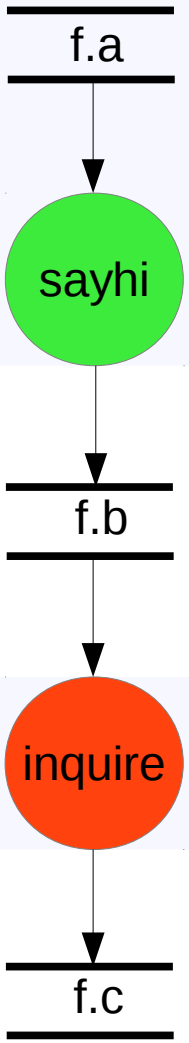
Step 2. Convert Workflow to DAX using the Python API




```
import os
from Pegasus.DAX3 import *

sayhipath = '/apps/pegtut/current/bin/sayhi.sh'
inquirepath = '/apps/pegtut/current/bin/inquire.sh'


# create an abstract DAX
dax = ADAG("sayhi_inquire")
```

How does Pegasus Work?

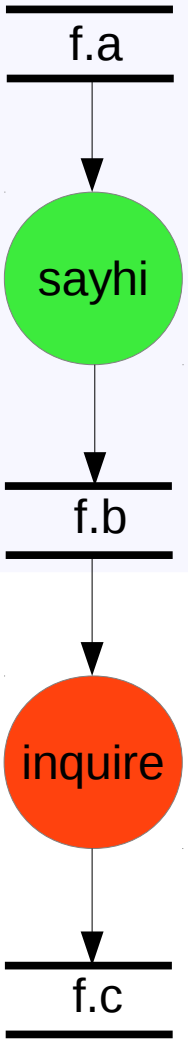
Step 2. Convert Workflow to DAX – Declare files and executables to replica catalog




```
# Add input file to the DAX-level replica catalog
a = File("f.a")
a.addPFN(PFN("file://" + os.path.join(os.getcwd(),"f.a"), "local"))
dax.addFile(a)

# Add executables to the DAX-level replica catalog
e_sayhi = Executable(namespace="sayhi_inquire", \
                     name="sayhi", version="1.0", \
                     os="linux", arch="x86_64", \
                     installed=False)
e_sayhi.addPFN(PFN("file://" + sayhipath, "condorpool"))
dax.addExecutable(e_sayhi)

e_inquire = Executable(namespace="sayhi_inquire", \
                       name="inquire", version="1.0", \
                       os="linux", arch="x86_64",
                       installed=False)
e_inquire.addPFN(PFN("file://" + inquirepath, "condorpool"))
dax.addExecutable(e_inquire)
```

How does Pegasus Work?

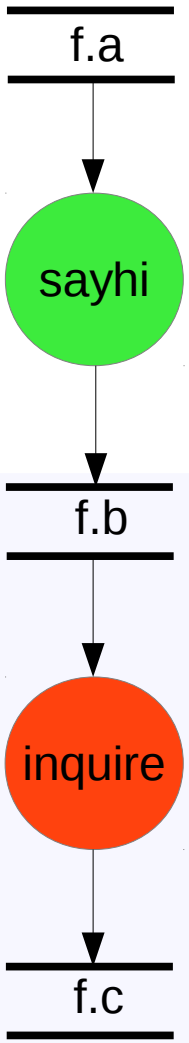
Step 2. Convert Workflow to DAX – Add jobs to the DAX




```
# Add the sayhi job
sayhi = Job(namespace="sayhi_inquire", \
            name="sayhi", version="1.0")
sayhi.addArguments('f.a')
b = File("f.b")
sayhi.uses(a, link=Link.INPUT)
sayhi.uses(b, link=Link.OUTPUT)
dax.addJob(sayhi)

# Add the inquire job (depends on the sayhi job)
inquire = Job(namespace="sayhi_inquire", \
              name="inquire", version="1.0")
inquire.addArguments('f.b')
c = File("f.c")
inquire.uses(b, link=Link.INPUT)
inquire.uses(c, link=Link.OUTPUT)
dax.addJob(inquire)
```

How does Pegasus Work?

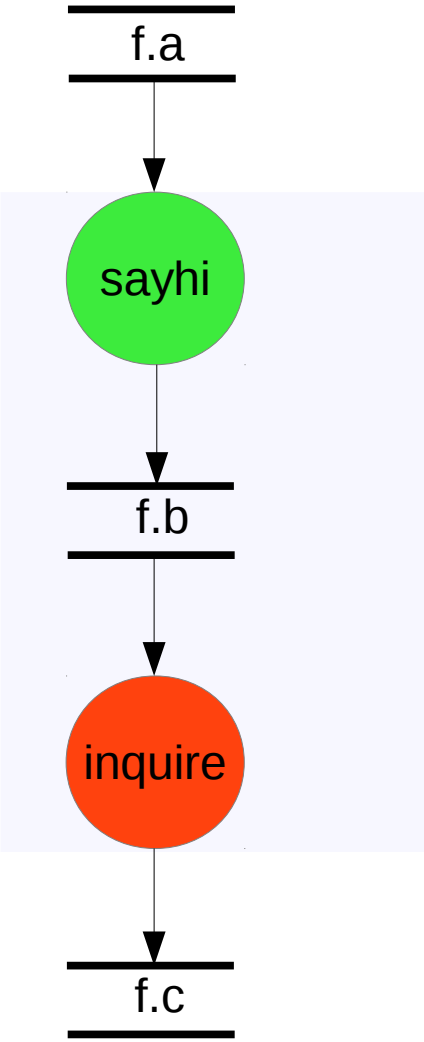
Step 2. Convert Workflow to DAX – Add jobs to the DAX




```
# Add the sayhi job
sayhi = Job(namespace="sayhi_inquire", \
            name="sayhi", version="1.0")
sayhi.addArguments('f.a')
b = File("f.b")
sayhi.uses(a, link=Link.INPUT)
sayhi.uses(b, link=Link.OUTPUT)
dax.addJob(sayhi)

# Add the inquire job (depends on the sayhi job)
inquire = Job(namespace="sayhi_inquire", \
              name="inquire", version="1.0")
inquire.addArguments('f.b')
c = File("f.c")
inquire.uses(b, link=Link.INPUT)
inquire.uses(c, link=Link.OUTPUT)
dax.addJob(inquire)
```

How does Pegasus Work?

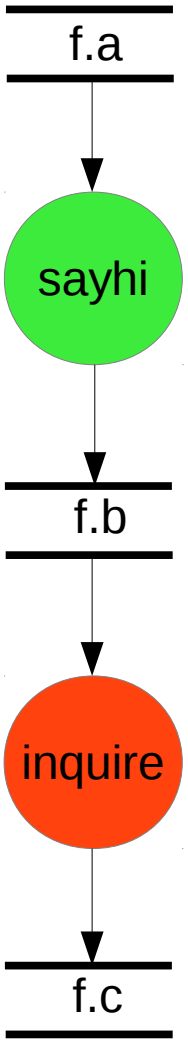
Step 2. Convert Workflow to DAX – Add control-flow dependencies




```
# Add control-flow dependencies
dax.addDependency(Dependency(parent=sayhi, child=inquire))
```

How does Pegasus Work?

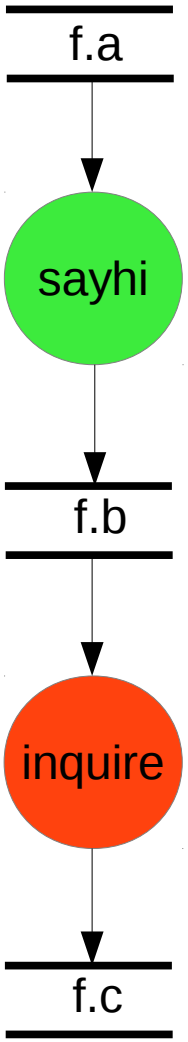
Step 2. Convert Workflow to DAX – Write DAX to file




```
# Write the DAX to file
with open('sayhiinquire.dax','w') as fp:
    dax.writeXML(fp)
```

Running the DAX

Step 3. Convert Workflow to DAX – Write DAX to file



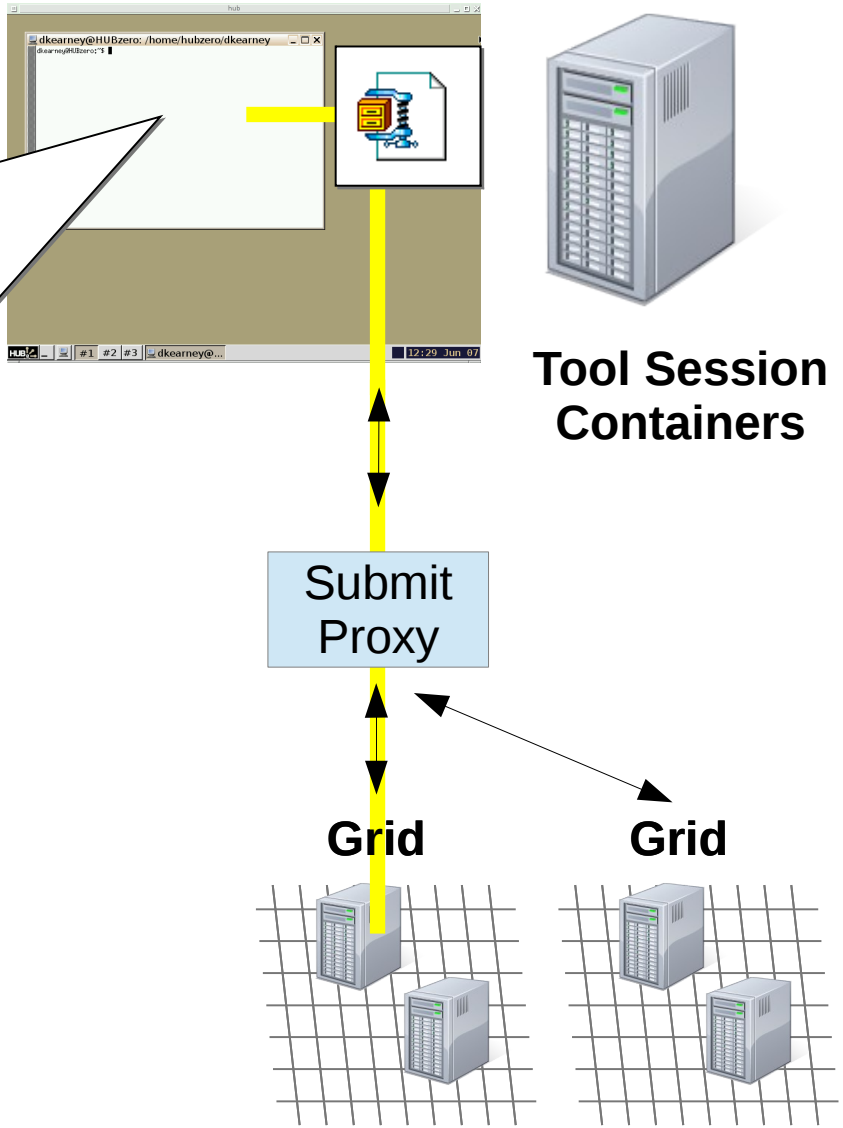

```
$ submit pegasus-plan --dax sayhiinquire.dax
```

Running The DAX

User's Workspace Terminal

```
$ submit pegasus-plan --dax sayhiinquire.dax

(989.0) Job Submitted at WF-DiaGrid
(989.0) DAG Running at WF-DiaGrid
…
(989.0) DAG Done at WF-DiaGrid

$ cat f.b

Hello pete!

$ cat f.c

Hello pete! How are you?
```

HUBzero Infrastructure

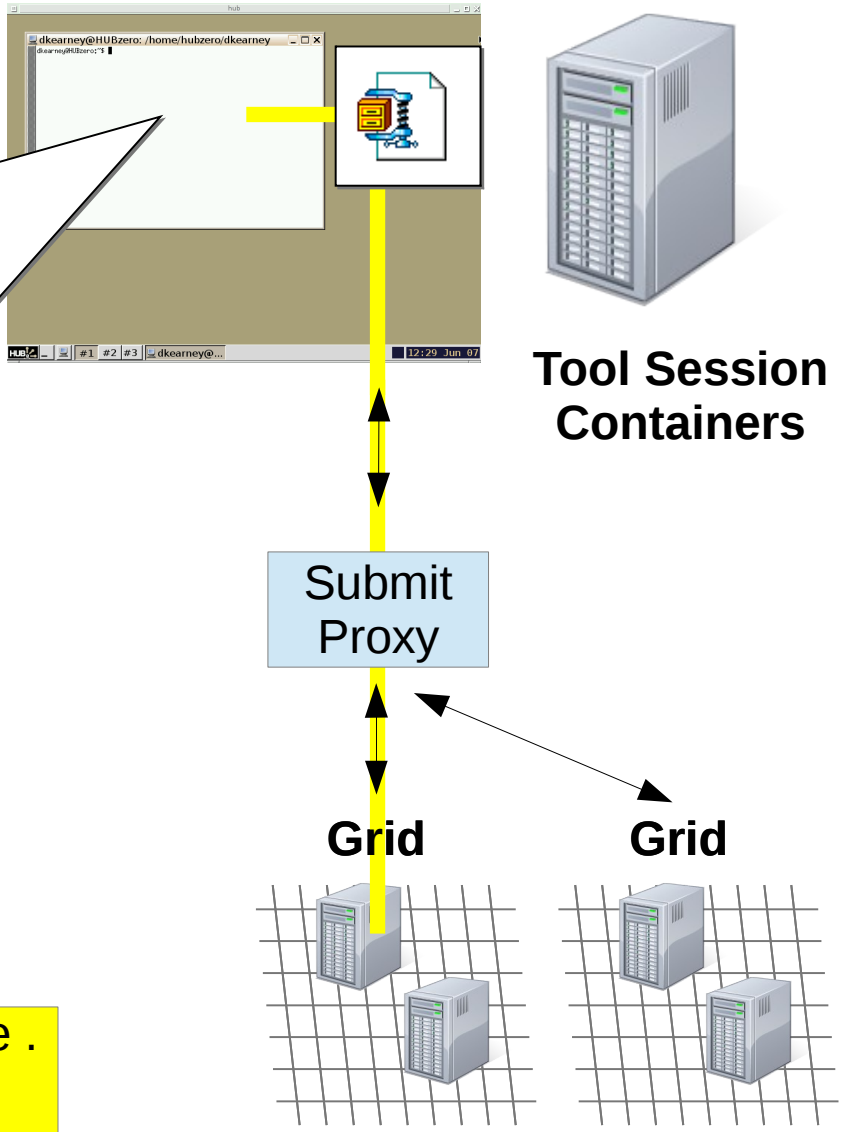


Running The DAX

User's Workspace Terminal

```
$ submit pegasus-plan --dax sayhiinquire.dax

(989.0) Job Submitted at WF-DiaGrid
(989.0) DAG Running at WF-DiaGrid
…
(989.0) DAG Done at WF-DiaGrid

$ cat f.b

Hello pete!

$ cat f.c

Hello pete! How are you?
```

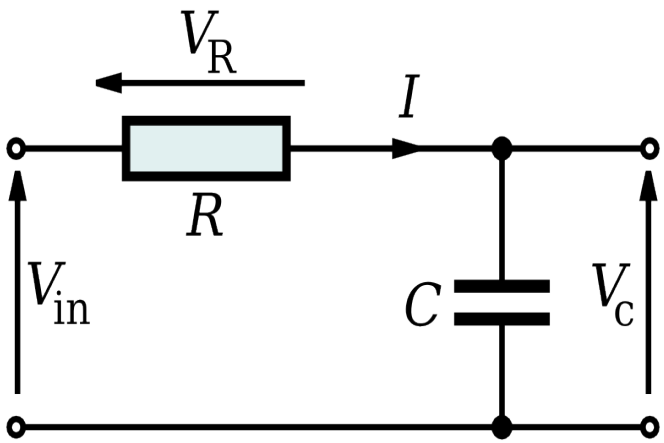
HUBzero Infrastructure



Try creating and running a DAX:

```
$ cp -r /apps/pegtut/current/examples/sayhi_inquire .
$ cd sayhi_inquire
$ ./createdax.py
$ submit pegasus-plan –dax sayhiinquire.dax
```

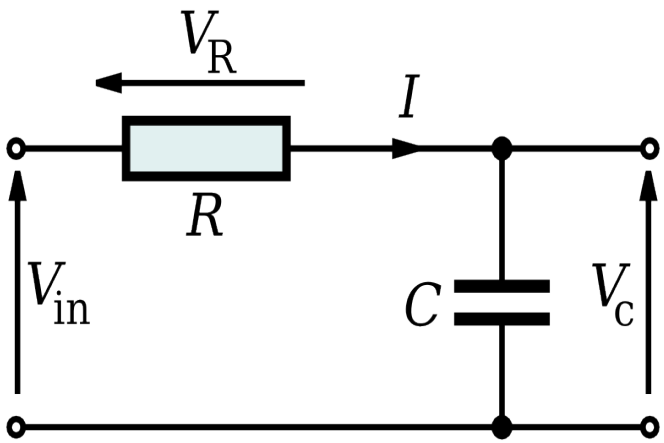
Sweeping Variables



Vc(t)= Vin*(1-exp(-t/(R*C)))

```
def Vc(Vin,R,C,lowerbound,upperbound):

    # store all solutions
    s = {'t':[],'v(t)':[]}

    for t in frange(lowerbound,upperbound,(upperbound-lowerbound)/100.0):
        v = Vin*(1-exp(-t/(R*C)))
        s['t'].append(t)
        s['v(t)'].append(v)

    return s
```

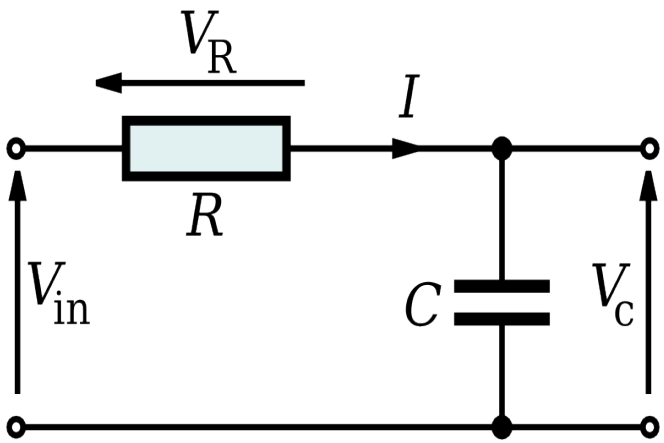
Sweeping Variables



Vc(t)= Vin*(1-exp(-t/(R*C)))

```
$ ls /apps/pegtut/current/examples/sim1.py --help

Usage: sim1.py [options]

Options:
  -h, --help            show this help message and exit
  --Vin=VIN             Input voltage to the system with units of volts (V)
  --R=R                 Impedance of the resistor with units of ohms
  --C=C                 Capacitance of the capacitor with units of farads (F)
  --lowerbound=LB  Lower bound of the time frame to examine in seconds (s)
  --upperbound=UB Upper bound of the time frame to examine in seconds (s)
  --log=LOG             Name of the file used to store results
```

Sweeping Variables



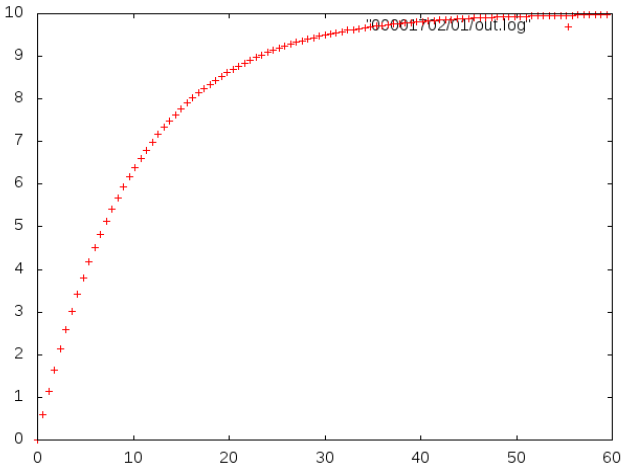
Vc(t)= Vin*(1-exp(-t/(R*C)))

$ /apps/pegtut/current/examples/sim1.py

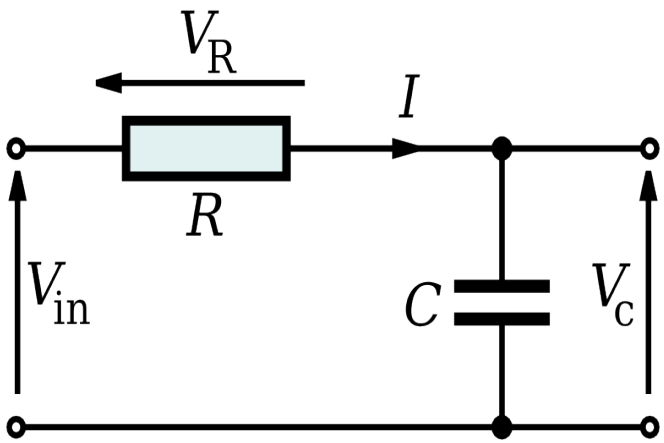
$ /apps/pegtut/current/examples/sim1.py --Vin 10

$ /apps/pegtut/current/examples/sim1.py --R 100e3

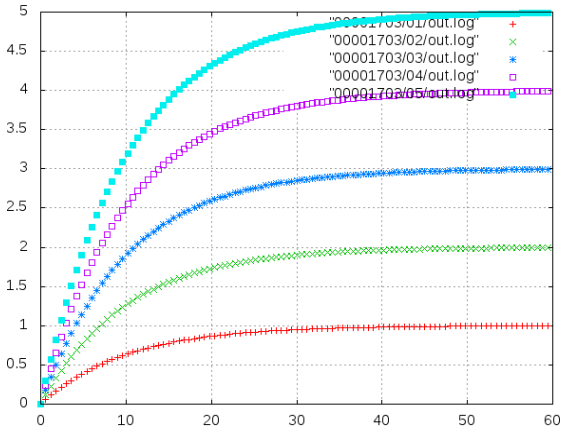
$ /apps/pegtut/current/examples/sim1.py –C 100e-6



Sweeping Variables

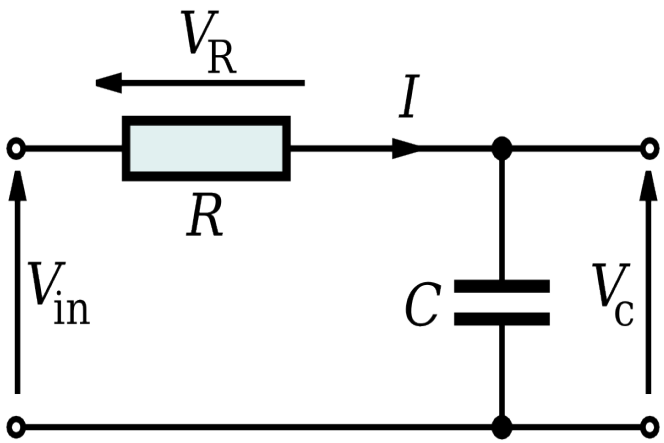


Vc(t)= Vin*(1-exp(-t/(R*C)))



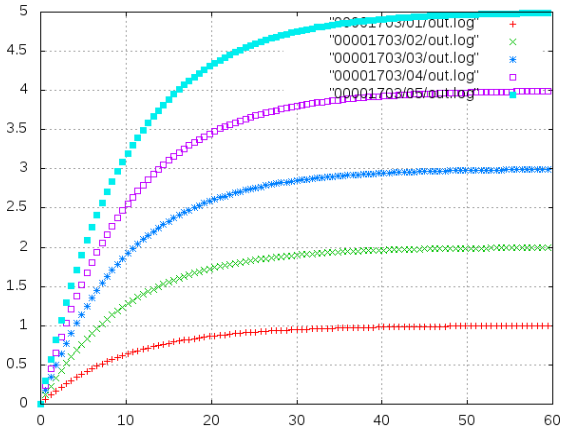

```
$ /apps/pegtut/current/examples/sim1.py

$ /apps/pegtut/current/examples/sim1.py --Vin 1

$ /apps/pegtut/current/examples/sim1.py --Vin 2

$ /apps/pegtut/current/examples/sim1.py --Vin 3

$ /apps/pegtut/current/examples/sim1.py --Vin 4

$ /apps/pegtut/current/examples/sim1.py --Vin 5
```

Sweeping Variables

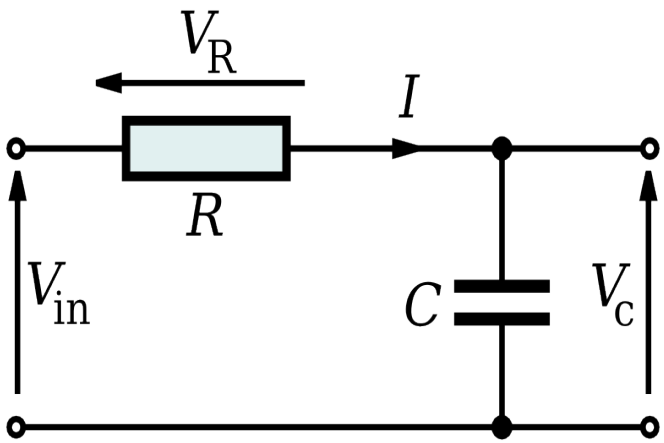


Vc(t)= Vin*(1-exp(-t/(R*C)))

$ submit -p @@Vin=1,2,3,4,5 ./sim1.py --Vin @@Vin

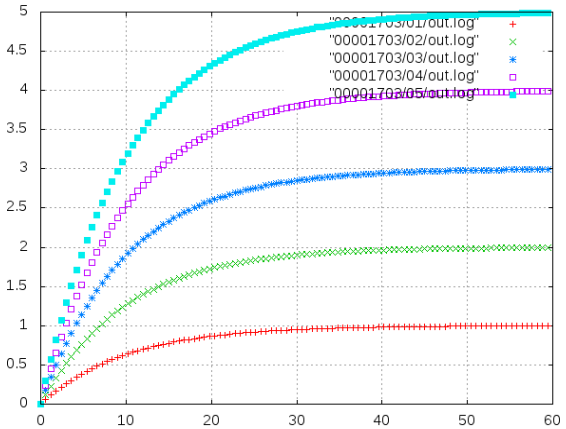


Sweeping Variables

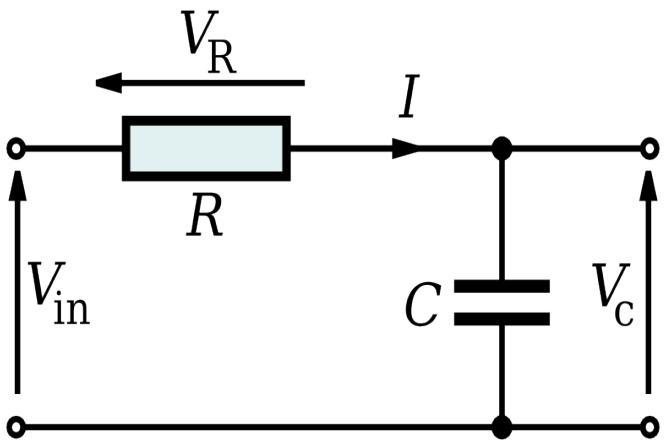


Vc(t)= Vin*(1-exp(-t/(R*C)))

$ submit -p @@Vin=1-5 ./sim1.py --Vin @@Vin

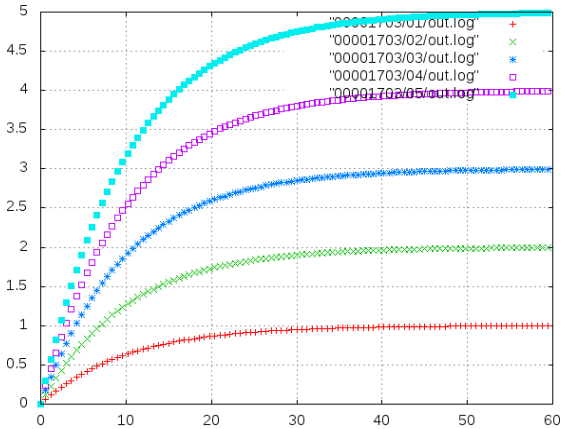


Sweeping Variables

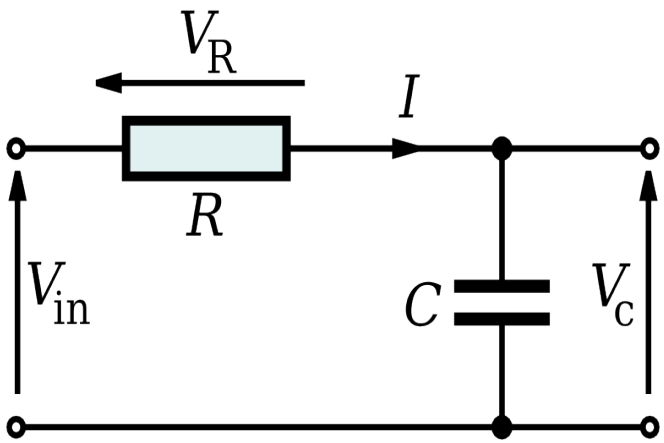


Vc(t)= Vin*(1-exp(-t/(R*C)))

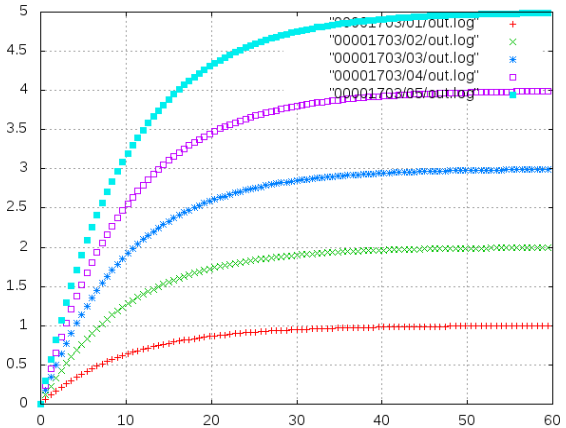
$ submit -p @@Vin=1:1:5 ./sim1.py --Vin @@Vin



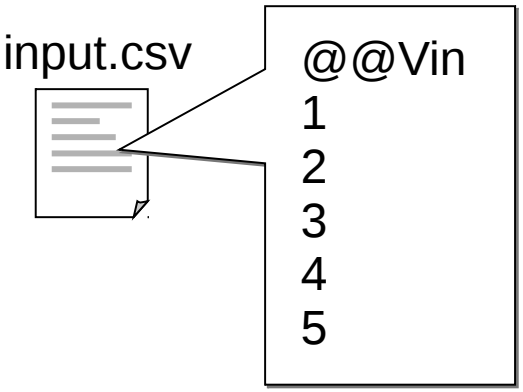
Sweeping Variables



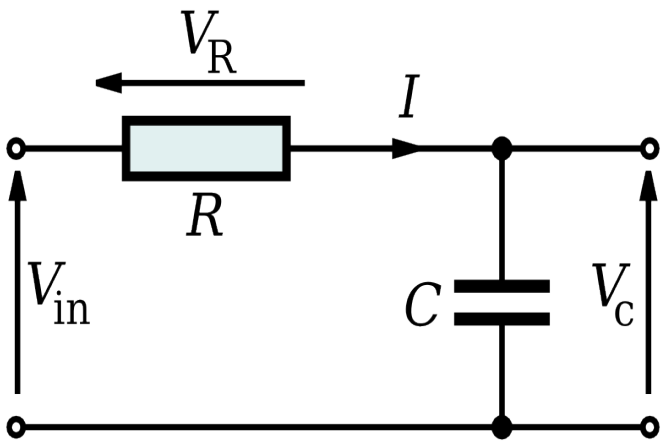
Vc(t)= Vin*(1-exp(-t/(R*C)))



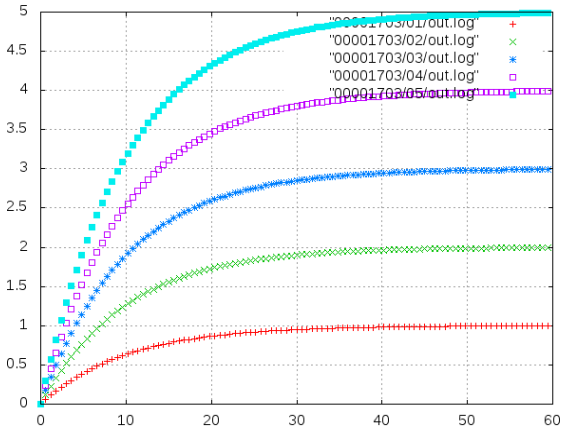
$ submit -d input.csv ./sim1.py --Vin @@Vin



Sweeping Variables



Vc(t)= Vin*(1-exp(-t/(R*C)))

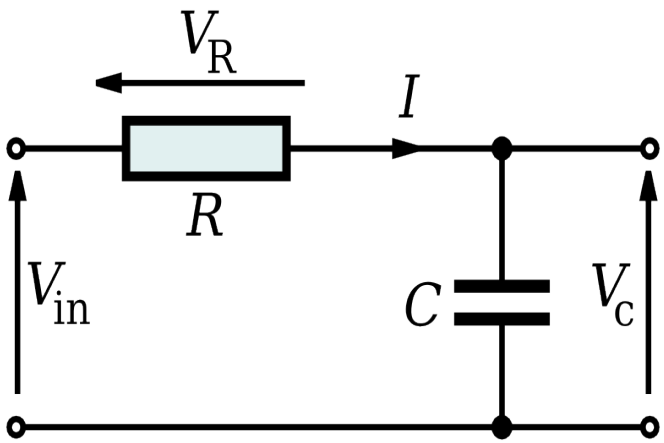


$ submit -d input.csv ./sim1.py --Vin @@Vin –C @@C

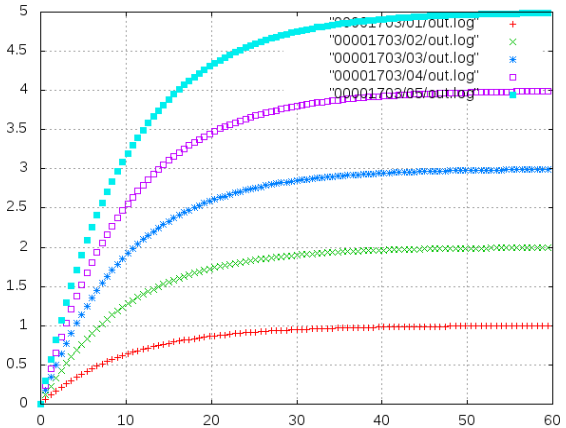
input.csv

```
@@Vin,@@C
1, 100e-6
2, 100e-6
3, 100e-6
4, 100e-6
5, 100e-6
```

Sweeping Variables



Vc(t)= Vin*(1-exp(-t/(R*C)))

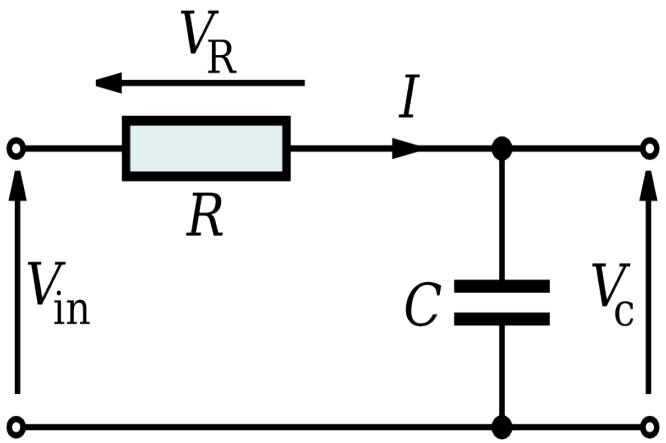


$ submit -p params ./sim1.py --Vin @@Vin –C @@C

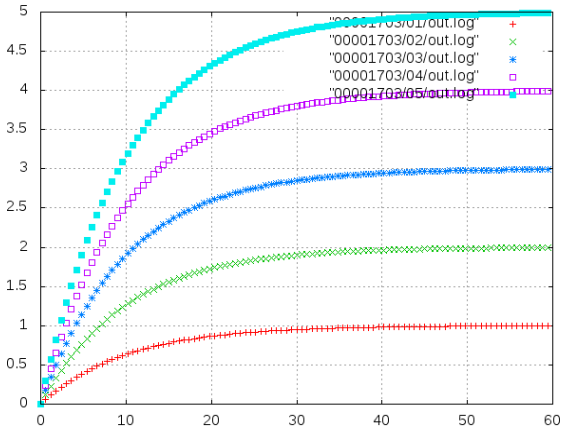
params

parameter @@Vin=1:1:5
parameter @@C=100e-6

Sweeping Variables



Vc(t)= Vin*(1-exp(-t/(R*C)))




```
$ cp -r /apps/pegtut/current/examples/capacitor_voltage .
$ cd capacitor_voltage
$ submit --local -p @@Vin=1,2,3,4,5 ./sim1.py --Vin @@Vin
$ submit --local -p @@Vin=1-5 ./sim1.py --Vin @@Vin
$ submit --local -p @@Vin=1:1:5 ./sim1.py --Vin @@Vin
$ submit --local -d input.csv ./sim1.py --Vin @@Vin
$ submit --local -p params ./sim1.py --Vin @@Vin
$ submit --local -p @@Vin=1-10 -p @@C=100e-6,100e-5 \
      ./sim1.py --Vin @@Vin --C @@C

# Check your plots quickly with "./plotsweep.py <job-directory>" :
$ ./plotsweep.py 00001720
```