University of Zurich[UZH]

High Performance Computing
Lecture 2

Douglas Potter
Cesare Cozza
Mark Eberlein

Last Week

- What is HPC?
- What is a Supercomputer?
- Connecting to Piz Daint & Eiger
- SSH & X11
- Editors
- Unix / Linux

Today's Topics

- Revision Control Systems
- How compilers work
- Eiger "Environment"
- Batch Queues
- Processes & Threads
- Compiling and running code

Online Resources

http://goalkicker.com/

- Git Notes for Professionals
- Linux Notes for Professionals
- Bash Notes for Professionals
- C Notes for Professionals

GNU Make

- GNU Autoconf
- CMake

CSCS User Portal

- SLURM Jobscript Generator

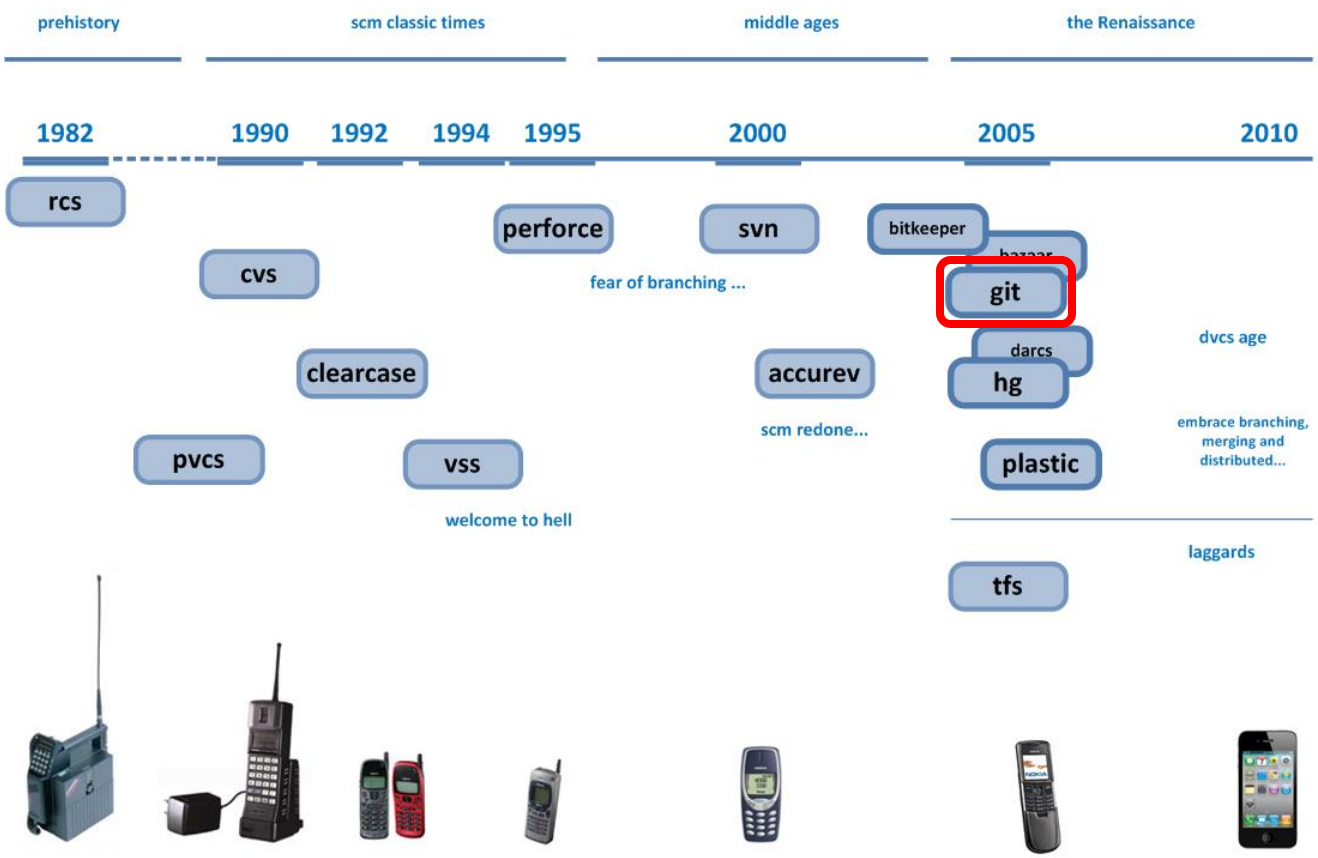
GitHub

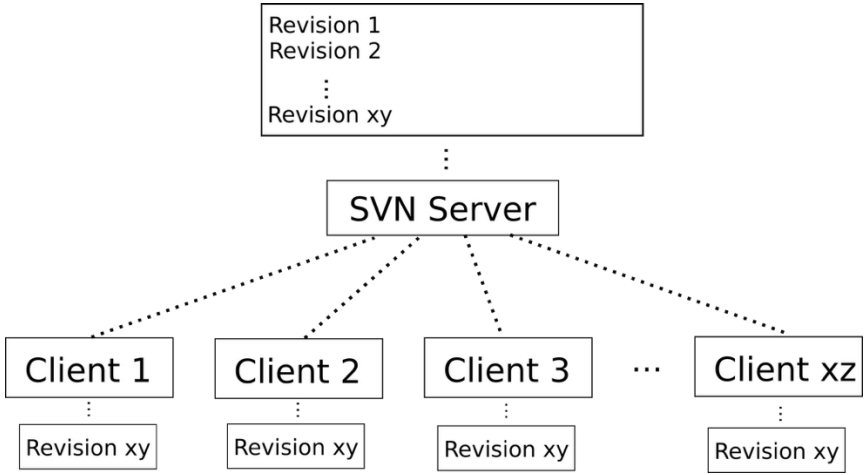
Source Code Management

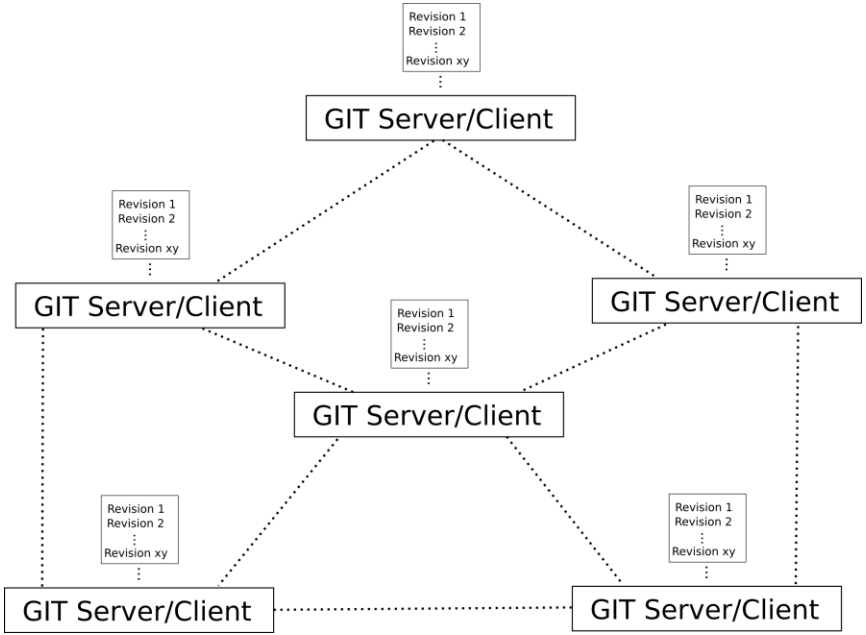
What/Why of Revision Control

- You change something and now it’s broken
- You forgot what you changed
- You start something and want to go back
- You want to know exactly what was changed
- You want to know who changed something
- You want to “undo” an old change
- You accidentally made a horrible mistake
- You want to work in parallel (different versions)
- You want to work as part of a team

SCM History


prehistory
scm classic times
middle ages
the Renaissance
1982
1990
1992
1994
1995
2000
2005
2010
rcs
cvs
clearcase
pvcs
vss
perforce
svn
bitkeeper
bazaar
git
darcs
hg
accurev
plastic
tfs
fear of branching ...
scm redone...
welcome to hell
dvcs age
embrace branching, merging and distributed...
laggards

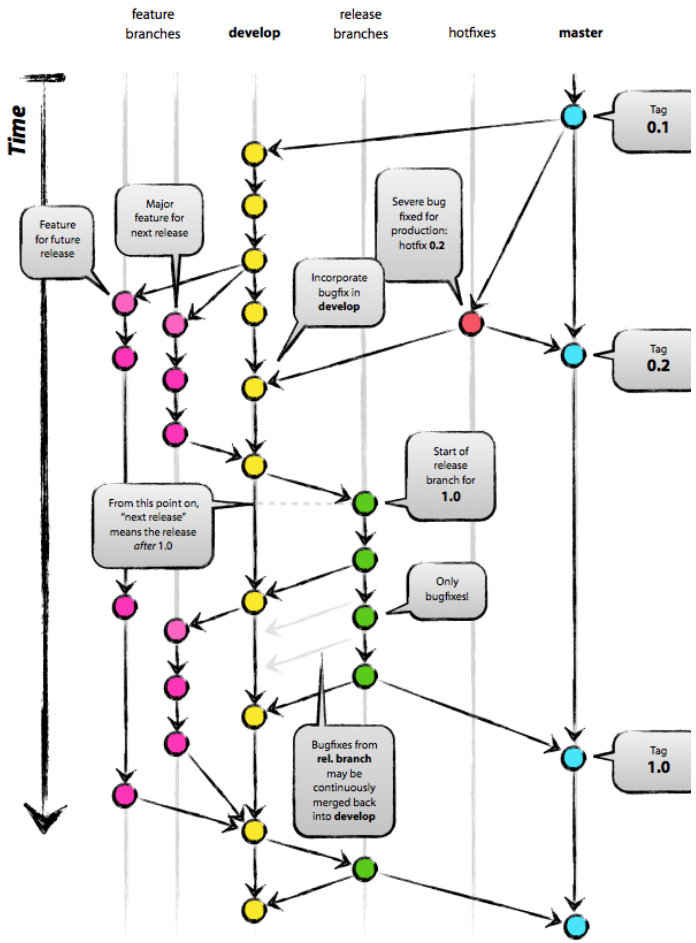




Git versus Subversion

Branches

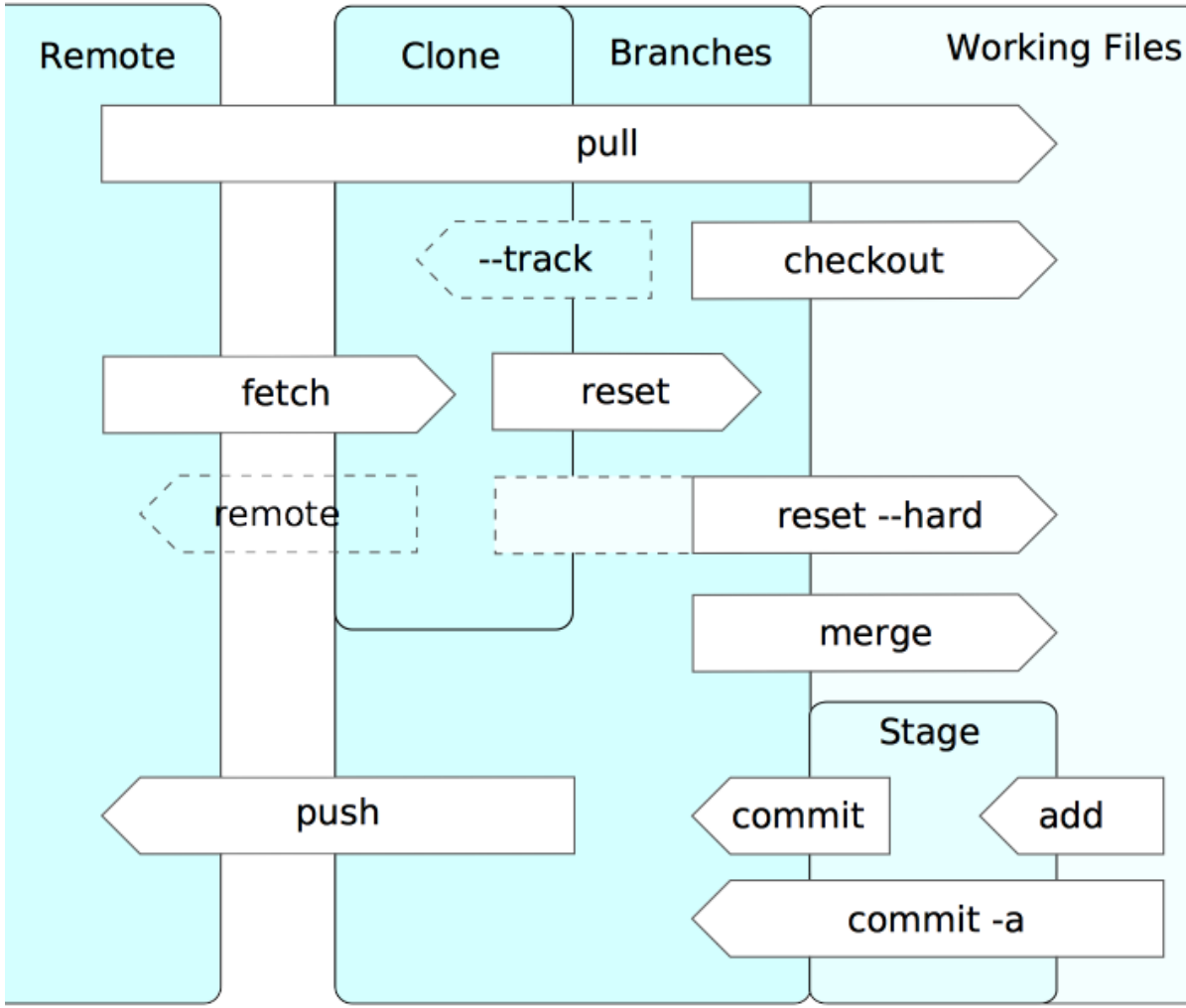
- git-flow (both an extension and a philosophy)
- “master” (“main”) branch – always stable working code
- “develop” branch – work in progress (waiting release)
- “feature” branches – new features you are working on
- “release” branches – versions that end up in “master”



University of Zurich[UZH]

Basic Git Operations

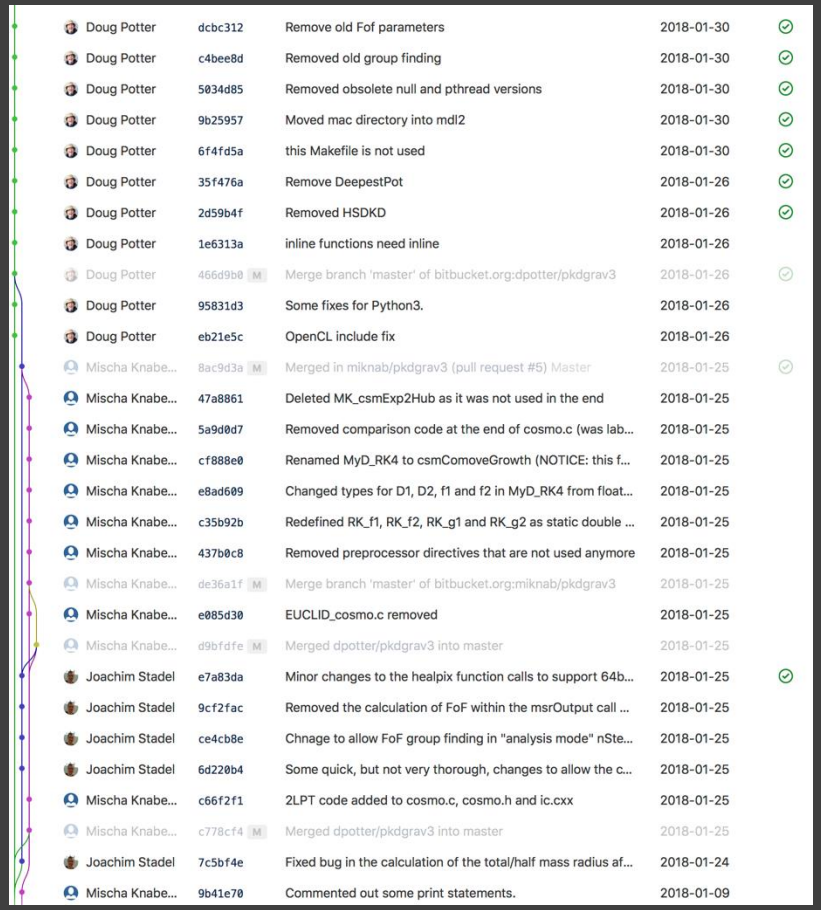
- “Working Files” are what you are probably used to.
- Clone/Branches are a local copy of the complete history.
- Remote
  - github.com
  - bitbucket.org
  - gitlab.uzh.ch
  - Or any other computer (or directory even!)



Almost Linear Development

Author	Commit	Message	Date	Builds
Doug Potter	4697bf1	Add openpa COPYRIGHT file	2018-02-14	✓
Joachim Stadel	1dfc65e	Went one step too far in the loop since D1[i+1] was set!	2018-02-13	✓
Doug Potter	1a4269e	MSC macros only under MSC	2018-02-13	✓
Doug Potter	58292c7	Now should compile under native Windows	2018-02-13	✓
Doug Potter	c222b31	More native windows compilation work (still in progress)	2018-02-13	✓
Doug Potter	191fb00	You cannot use a variable for size of local variables	2018-02-13	✓
Doug Potter	ab78c44	Why was mpi included here?	2018-02-13	✓
Doug Potter	71014b5 M	merge	2018-02-12	✓
Doug Potter	edff8a7	Some updates to support native windows	2018-02-12	
Doug Potter	dfe9661	Start of integrating base CUDA into MDL	2018-02-08	✓
Doug Potter	5d62d1c	Set sensible defaults when present	2018-02-07	✓
Doug Potter	0062ef1	Simple Windows instructions	2018-02-07	✓
Doug Potter	f1c8375	Add pkdgrav3 to the installation	2018-02-07	
Doug Potter	fc06662	Compilation on Windows	2018-02-07	✓
Doug Potter	c262c4f	A little to overzealous with the separation (for now)	2018-02-06	✓
Doug Potter	a6b5eeb	Separate generic CUDA routines better	2018-02-06	✓
Doug Potter	69fb03b	compilation fixes for Cray under cmake	2018-02-06	✓
Doug Potter	76acd63	Fix string literal problems	2018-02-06	
Doug Potter	4b8ce39	cmake should now work (perhaps everywhere)	2018-02-06	✓
Doug Potter	607ba6d	Preliminary cmake support (not yet completely ready)	2018-02-06	✓
Doug Potter	0fe6d50	Remove null/pthread build targets (there weren't used)	2018-02-05	✓
Doug Potter	63917ff	Features/macros cleanup	2018-02-05	✓
Doug Potter	1159819	Removed COOLING and Fortran support	2018-02-05	✓
Doug Potter	ab9106f	CHANGESOFT is not used	2018-02-05	✓
Doug Potter	82e60f0	Removed global group assignment code. This will be done diff...	2018-02-02	✓
Doug Potter	c60199c	Removed nLowTot/nHighTot from the pst; they were never used.	2018-02-02	✓
Doug Potter	f1d372f	Make argv processing optional by passing NULL	2018-02-01	✓
Doug Potter	a967fb1	mdl2 can now be compiled stand-alone with cmake	2018-01-31	✓

Parallel Development



Doug Potter	dcbc312	Remove old Fof parameters	2018-01-30
Doug Potter	c4bee8d	Removed old group finding	2018-01-30
Doug Potter	5034d85	Removed obsolete null and pthread versions	2018-01-30
Doug Potter	9b25957	Moved mac directory into mdl2	2018-01-30
Doug Potter	6f4fd5a	this Makefile is not used	2018-01-30
Doug Potter	35f476a	Remove DeepestPot	2018-01-26
Doug Potter	2d59b4f	Removed HSDKD	2018-01-26
Doug Potter	1e6313a	inline functions need inline	2018-01-26
Doug Potter	466d9b0 M	Merge branch 'master' of bitbucket.org:dpotter/pkdgrav3	2018-01-26
Doug Potter	95831d3	Some fixes for Python3.	2018-01-26
Doug Potter	eb21e5c	OpenCL include fix	2018-01-26
Mischa Knabe...	8ac9d3a M	Merged in miknab/pkdgrav3 (pull request #5) Master	2018-01-25
Mischa Knabe...	47a8861	Deleted MK_csmExp2Hub as it was not used in the end	2018-01-25
Mischa Knabe...	5a9d0d7	Removed comparison code at the end of cosmo.c (was lab...	2018-01-25
Mischa Knabe...	cf888e0	Renamed MyD_RK4 to csmComoveGrowth (NOTICE: this f...	2018-01-25
Mischa Knabe...	e8ad609	Changed types for D1, D2, f1 and f2 in MyD_RK4 from float...	2018-01-25
Mischa Knabe...	c35b92b	Redefined RK_f1, RK_f2, RK_g1 and RK_g2 as static double ...	2018-01-25
Mischa Knabe...	437b0c8	Removed preprocessor directives that are not used anymore	2018-01-25
Mischa Knabe...	de36a1f M	Merge branch 'master' of bitbucket.org:miknab/pkdgrav3	2018-01-25
Mischa Knabe...	e085d30	EUCLID_cosmo.c removed	2018-01-25
Mischa Knabe...	d9bfdfe M	Merged dpotter/pkdgrav3 into master	2018-01-25
Joachim Stadel	e7a83da	Minor changes to the healpix function calls to support 64b...	2018-01-25
Joachim Stadel	9cf2fac	Removed the calculation of FoF within the msrOutput call ...	2018-01-25
Joachim Stadel	ce4cb8e	Chnage to allow FoF group finding in "analysis mode" nSte...	2018-01-25
Joachim Stadel	6d220b4	Some quick, but not very thorough, changes to allow the c...	2018-01-25
Mischa Knabe...	c66f2f1	2LPT code added to cosmo.c, cosmo.h and ic.cxx	2018-01-25
Mischa Knabe...	c778cf4 M	Merged dpotter/pkdgrav3 into master	2018-01-25
Joachim Stadel	7c5bf4e	Fixed bug in the calculation of the total/half mass radius af...	2018-01-24
Mischa Knabe...	9b41e70	Commented out some print statements.	2018-01-09

GIT Example

```
dpotter@daint103:~> cat .gitconfig            (Set this first!)
[user]
        email = douglas.potter@uzh.ch
        name = Doug Potter
dpotter@daint103:~> cd cpi
dpotter@daint103:~/cpi> ls
cpi_mpi.c  cpi_openmp.c  Makefile
dpotter@daint103:~/cpi> git init              (Done only once)
Initialized empty Git repository in /users/dpotter/cpi/.git/
dpotter@daint103:~/cpi> git status         (Output abbreviated)
Untracked files:
        Makefile
        cpi_mpi.c
        cpi_openmp.c
dpotter@daint103:~/cpi> git add Makefile cpi_mpi.c cpi_openmp.c
dpotter@daint103:~/cpi> git commit -m "Initial import"
[master (root-commit) b0f821d] Initial import
 3 files changed, 145 insertions(+)
 create mode 100644 Makefile
 create mode 100644 cpi_mpi.c
 create mode 100644 cpi_openmp.c
```

Compilers

C & Fortran

“Hello World” in C

```
#include <stdio.h>

int main(int argc,char *argv[]) {
    printf("Hello world\n");
    return 0;
    }
```

“Hello World” in “Machine Code”

```
.code
main:    mova S0,r0
         call printf
         movi #0,r0
         ret

.data
S0:      .text “Hello world\n”
```

```c
int main(int argc,char *argv[]) {
    printf("Hello world\n");
    return 0;
    }
```

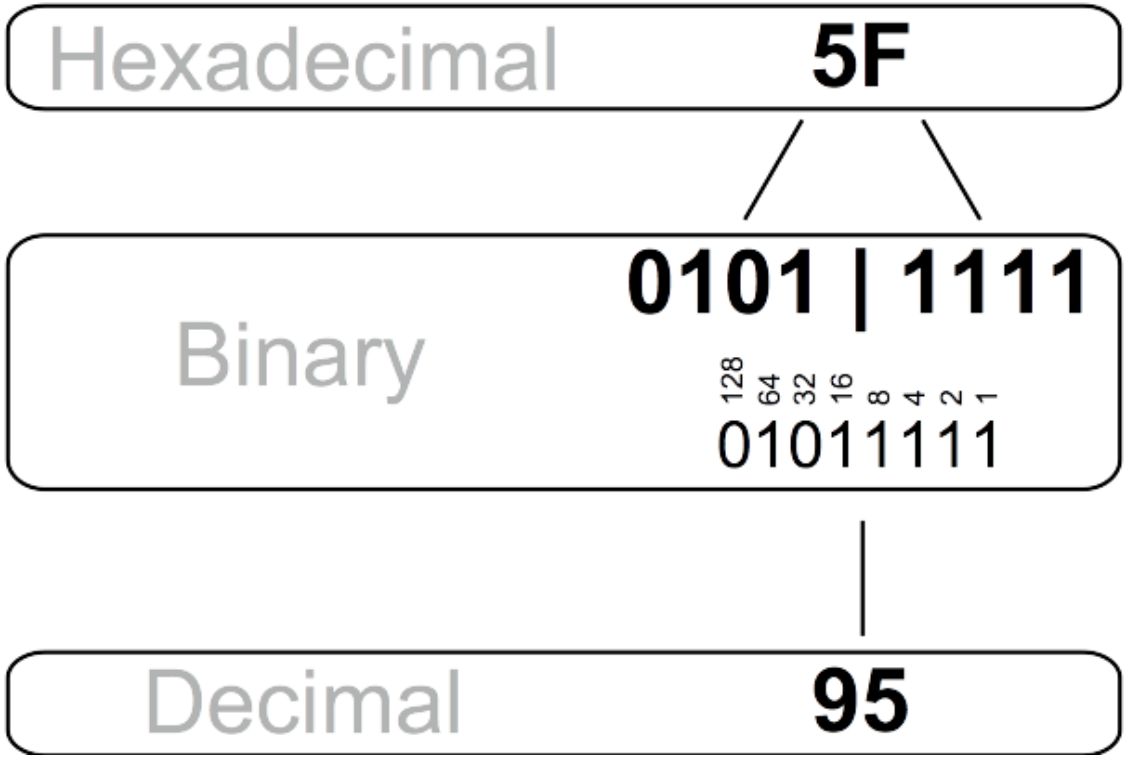
Real Assembler Output

```
        .section        __TEXT,__text,regular,pure_instructions
        .macosx_version_min 10, 13
        .globl          _main
        .p2align        4, 0x90
_main:                          ## @main
        .cfi_startproc
## BB#0:
        pushq           %rbp
Lcfi0:
        .cfi_def_cfa_offset 16
Lcfi1:
        .cfi_offset %rbp, -16
        movq            %rsp, %rbp
Lcfi2:
        .cfi_def_cfa_register %rbp
        leaq            L_str(%rip), %rdi
        callq           _puts
        xorl            %eax, %eax
        popq            %rbp
        retq
        .cfi_endproc

        .section        __TEXT,__cstring,cstring_literals
L_str:                          ## @str
        .asciz          "Hello world"


.subsections_via_symbols
```

```
int main(int argc,char *argv[]) {
    printf("Hello world\n");
    return 0;
    }
```

“man puts”

```
PUTS(3)        BSD Library Functions Manual        PUTS(3)

NAME
     puts -- output a line to a stream

LIBRARY
     Standard C Library (libc, -lc)

SYNOPSIS
     #include <stdio.h>
     int     puts(const char *s);

DESCRIPTION
     The function puts() writes the string s, and a
     terminating newline character to the stream stdout.
```


Hexadecimal
5F
Binary
0101 | 1111
128
64
32
16
8
4
2
1
01011111
Decimal
95

Binary & Hex

Boolean Logic

```
leaq L_str(%rip), %rdi
callq    _puts
xorl %eax, %eax
popq %rbp
retq
```

A	0	1	0	1
B	0	0	1	1

false	0	0	0	0
$A \wedge B \Leftrightarrow \bar{A} \not\leftarrow B \Leftrightarrow A \not\rightarrow \bar{B} \Leftrightarrow \bar{A} \downarrow \bar{B}$	0	0	0	1
$A \not\leftarrow B \Leftrightarrow \bar{A} \wedge B \Leftrightarrow A \downarrow \bar{B} \Leftrightarrow \bar{A} \not\rightarrow \bar{B}$	0	0	1	0
B	0	0	1	1
$A \not\rightarrow B \Leftrightarrow \bar{A} \downarrow B \Leftrightarrow A \wedge \bar{B} \Leftrightarrow \bar{A} \not\leftarrow \bar{B}$	0	1	0	0
A	0	1	0	1
$A \oplus B \Leftrightarrow \bar{A} \leftrightarrow B \Leftrightarrow A \leftrightarrow \bar{B} \Leftrightarrow \bar{A} \oplus \bar{B}$	0	1	1	0
$A \vee B \Leftrightarrow \bar{A} \rightarrow B \Leftrightarrow A \leftarrow \bar{B} \Leftrightarrow \bar{A} \uparrow \bar{B}$	0	1	1	1
$A \downarrow B \Leftrightarrow \bar{A} \not\rightarrow B \Leftrightarrow A \not\leftarrow \bar{B} \Leftrightarrow \bar{A} \wedge \bar{B}$	1	0	0	0
$A \leftrightarrow B \Leftrightarrow \bar{A} \oplus B \Leftrightarrow A \oplus \bar{B} \Leftrightarrow \bar{A} \leftrightarrow \bar{B}$	1	0	0	1
$\bar{A}$	1	0	1	0
$A \rightarrow B \Leftrightarrow \bar{A} \vee B \Leftrightarrow A \uparrow \bar{B} \Leftrightarrow \bar{A} \leftarrow \bar{B}$	1	0	1	1
$\bar{B}$	1	1	0	0
$A \leftarrow B \Leftrightarrow \bar{A} \uparrow B \Leftrightarrow A \vee \bar{B} \Leftrightarrow \bar{A} \rightarrow \bar{B}$	1	1	0	1
$A \uparrow B \Leftrightarrow \bar{A} \leftarrow B \Leftrightarrow A \rightarrow \bar{B} \Leftrightarrow \bar{A} \vee \bar{B}$	1	1	1	0
true	1	1	1	1

Simple Arithmetic Function (x86)

```
float f(float a,float b,
        float c,float d,
        float e){
  return (a-b)*c + d/e;
  }
```

```
Disassembly of section .text:
   0:  55              push   %rbp
   1:  48 89 e5        mov    %rsp,%rbp
   4:  f3 0f 5c c1     subss  %xmm1,%xmm0
   8:  f3 0f 59 c2     mulss  %xmm2,%xmm0
   c:  f3 0f 5e dc     divss  %xmm4,%xmm3
  10:  f3 0f 58 c3     addss  %xmm3,%xmm0
  14:  5d              pop    %rbp
  15:  c3              retq
```

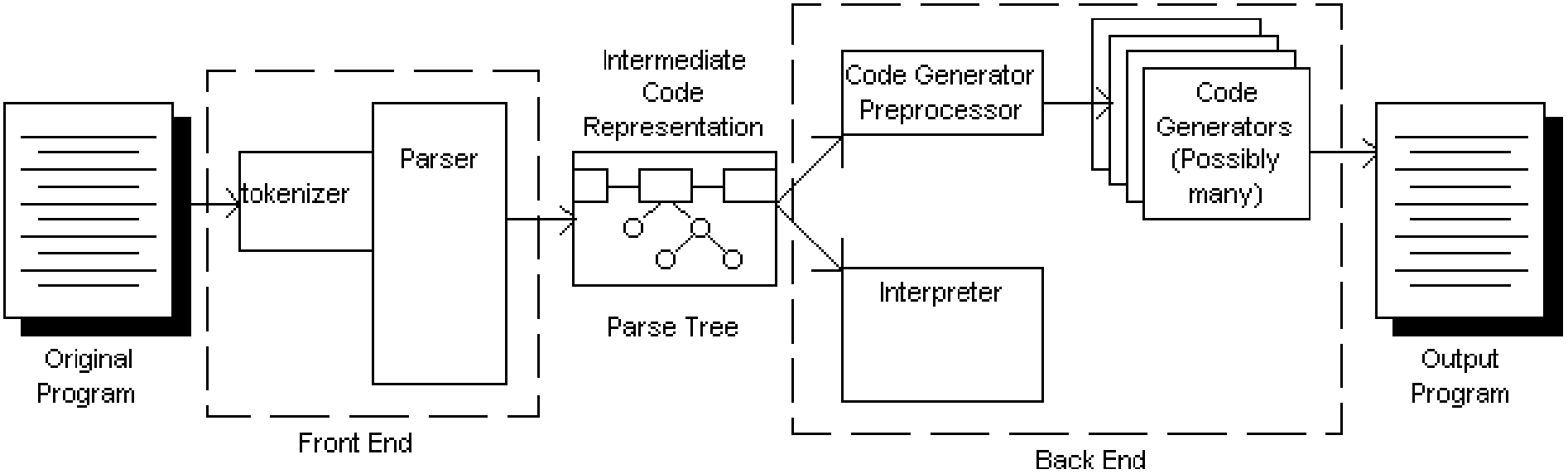
```
_f:
  pushq %rbp
  movq  %rsp, %rbp
  subss %xmm1, %xmm0
  mulss %xmm2, %xmm0
  divss %xmm4, %xmm3
  addss %xmm3, %xmm0
  popq  %rbp
  retq
```

gcc -O3 -S -c -o test.s test.c

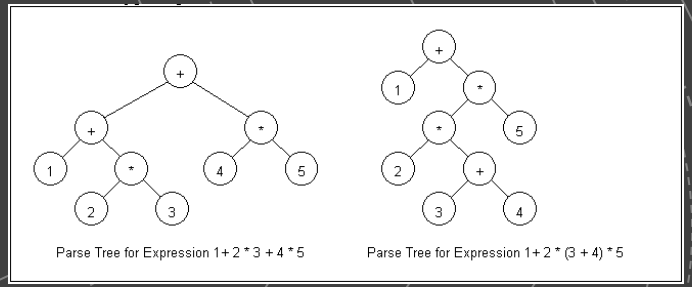
Integer Division (ARM)

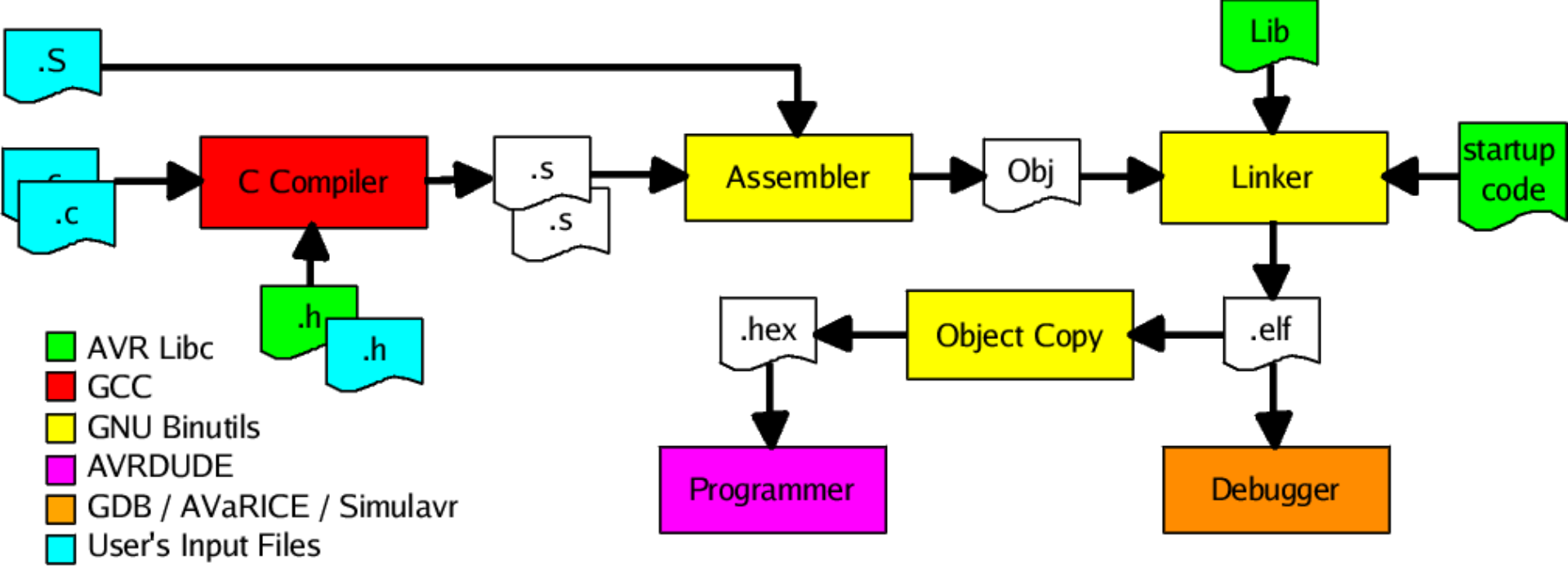
```
int div(int a, int b) {
    return a / b;
}

int div373487291(int a) {
    return a / 373487291;
}
```

```
_div:
    sdiv       w0, w0, w1
    ret


_div373487291:

    mov   w8, #16465   ; =0x4051
    movk  w8, #23551, lsl #16
    smull x8, w0, w8
    lsr   x9, x8, #63
    asr   x8, x8, #59
    add   w0, w8, w9
    ret
```



Anatomy of a Compiler





Compilation Process

“Modules” on Eiger

```
[dpotter@ela2 ~]$ ssh eiger
================================================================
       IMPORTANT NOTICE FOR USERS of Eiger.Alps

  Documentation: https://docs.cscs.ch/clusters/eiger
  Service Desk at https://support.cscs.ch
================================================================

[dpotter@eiger-ln004 ~]$ module load cray
[dpotter@eiger-ln004 ~]$ module swap PrgEnv-cray PrgEnv-gnu

Lmod is automatically replacing "cce/17.0.0" with "gcc-
native/12.3".


Due to MODULEPATH changes, the following have been reloaded:
  1) cray-libsci/23.12.5     2) cray-mpich/8.1.28

[dpotter@eiger-ln004 ~]$
```

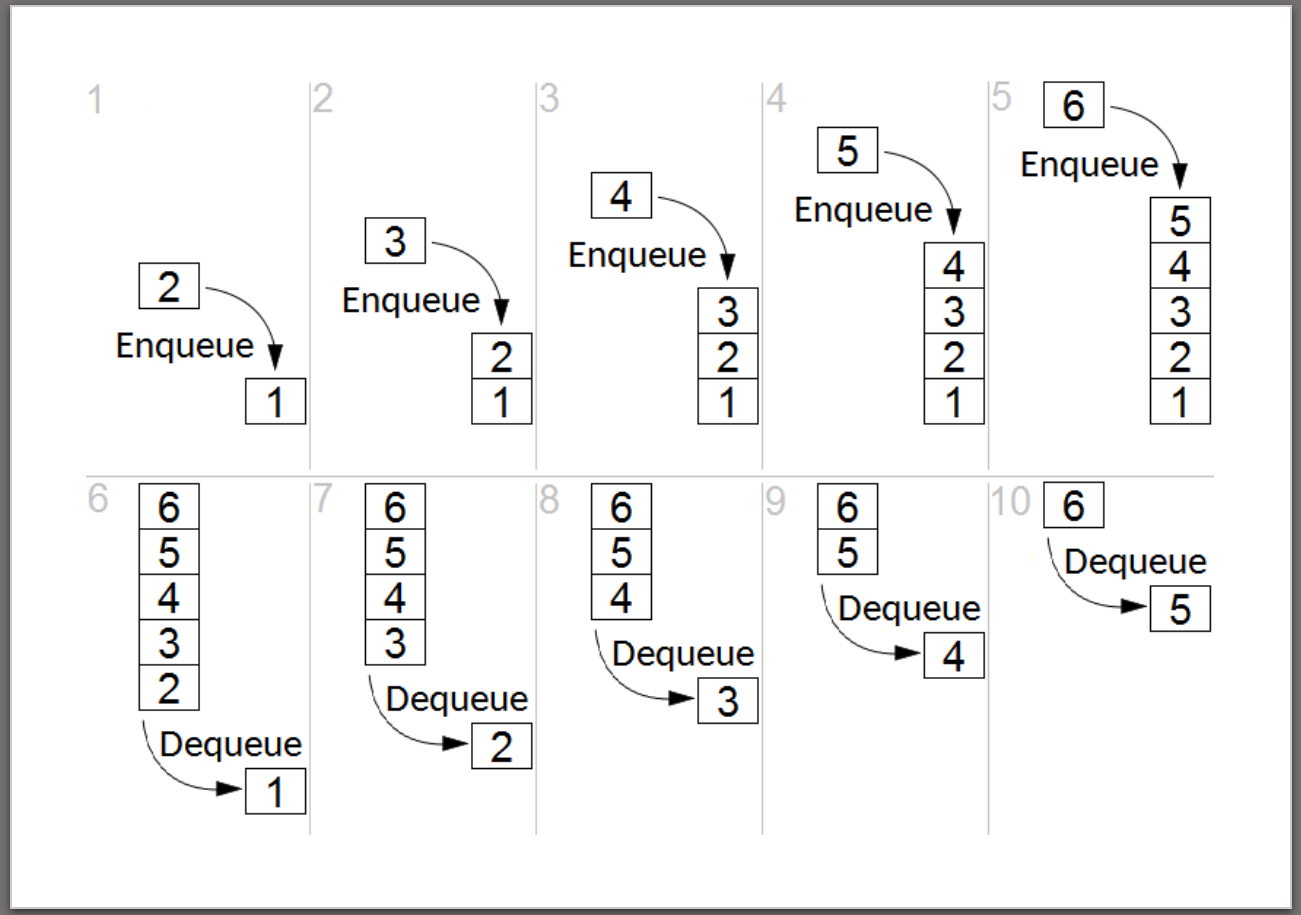
Compiler Suites

Cray Compiler (PrgEnv-cray)	Available on Cray Computers
GNU Compiler (PrgEnv-gnu)	Free for everyone
Intel Compiler (PrgEnv-intel)	Now free for everyone
Portland (PrgEnv-pgi)	Important for OpenACC (GPU)
Clang (Apple & Cray)	Open Source & GNU Compatible
Visual Studio (Windows)	Not usually High Performance

List Loaded Modules (Eiger)

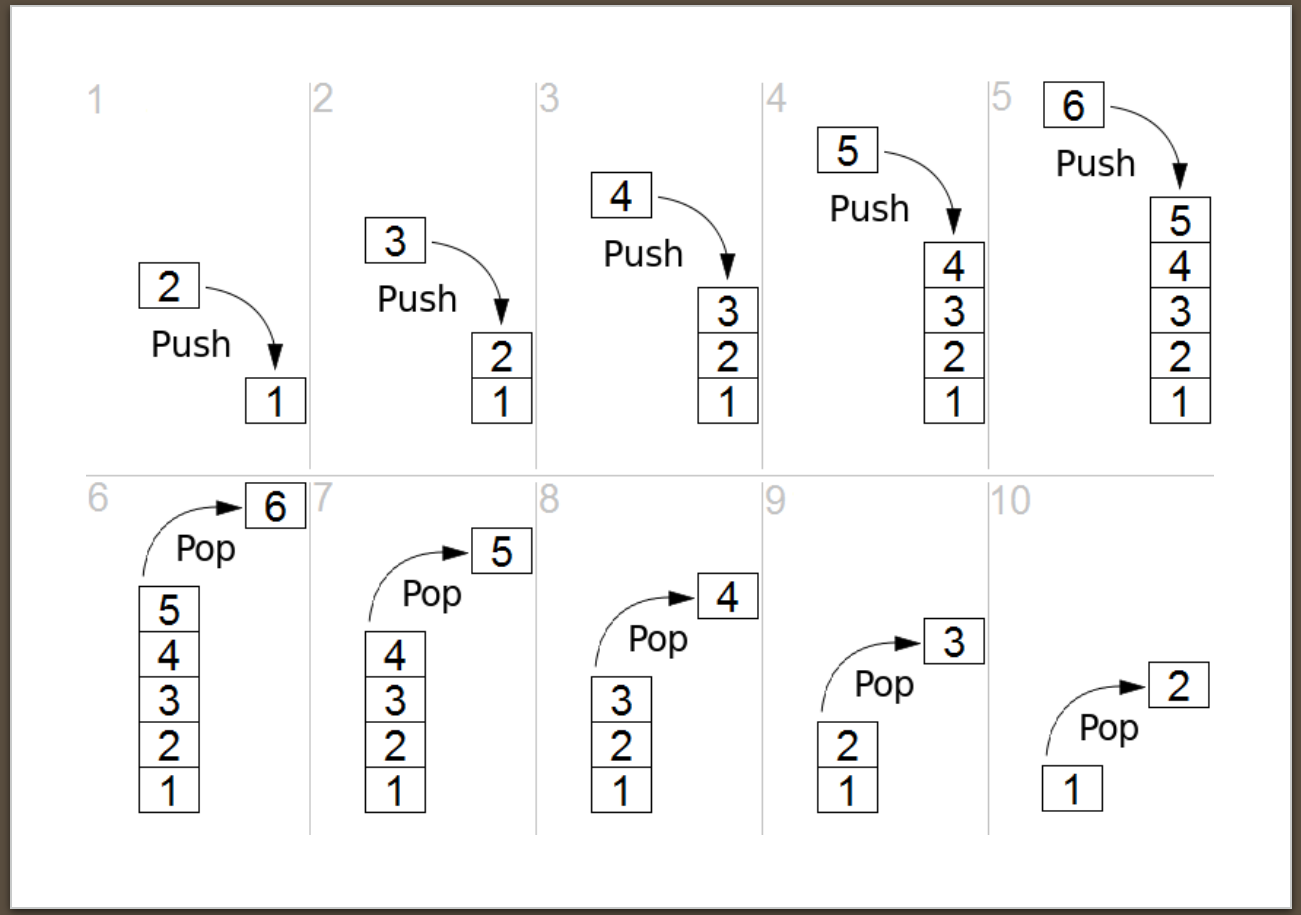
```
[eiger][dpotter@eiger-ln003 ~]$ module load cray
[eiger][dpotter@eiger-ln003 ~]$ module list

Currently Loaded Modules:
  1) craype-x86-rome      4) xpmem/2.8.2-1.0_3.9__g84a27a5.shasta   7) cray-dsmml/0.2.2     10) craype/2.7.30           13) cray/23.12
  2) libfabric/1.15.2.0   5) PrgEnv-cray/8.5.0                      8) cray-libsci/23.12.5  11) perftools-base/23.12.0
  3) craype-network-ofi   6) cce/17.0.0                             9) cray-mpich/8.1.28    12) cpe/23.12

[eiger][dpotter@eiger-ln003 ~]$ cc --version
Cray clang version 17.0.0  (b59b7a8e9169719529cf5ab440f3c301e515d047)
Target: x86_64-unknown-linux-gnu
Thread model: posix
InstalledDir: /opt/cray/pe/cce/17.0.0/cce-clang/x86_64/share/../bin

[eiger][dpotter@eiger-ln003 ~]$ module swap PrgEnv-cray PrgEnv-gnu

Lmod is automatically replacing "cce/17.0.0" with "gcc-native/12.3".


Due to MODULEPATH changes, the following have been reloaded:
  1) cray-libsci/23.12.5     2) cray-mpich/8.1.28

[eiger][dpotter@eiger-ln003 ~]$ module list

Currently Loaded Modules:
  1) craype-x86-rome      4) xpmem/2.8.2-1.0_3.9__g84a27a5.shasta   7) cray/23.12       10) cray-dsmml/0.2.2     13) PrgEnv-gnu/8.5.0
  2) libfabric/1.15.2.0   5) perftools-base/23.12.0                 8) gcc-native/12.3  11) cray-mpich/8.1.28
  3) craype-network-ofi   6) cpe/23.12                              9) craype/2.7.30    12) cray-libsci/23.12.5

[eiger][dpotter@eiger-ln003 ~]$ cc --version
gcc-12 (SUSE Linux) 12.3.0
Copyright (C) 2022 Free Software Foundation, Inc.
This is free software; see the source for copying conditions.  There is NO
warranty; not even for MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.
```

Batch Queue Systems



“Queue”

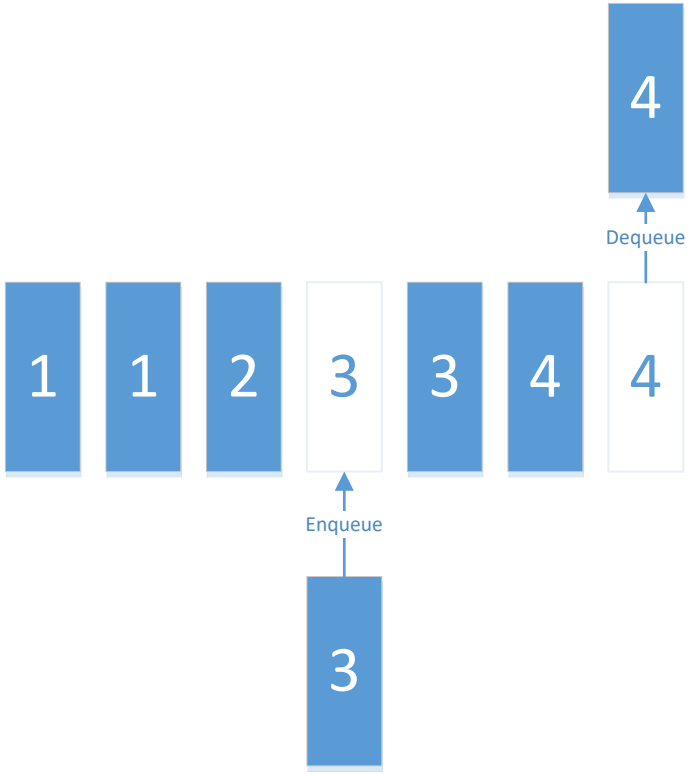
First In,
First Out
(FIFO)



"Stack"

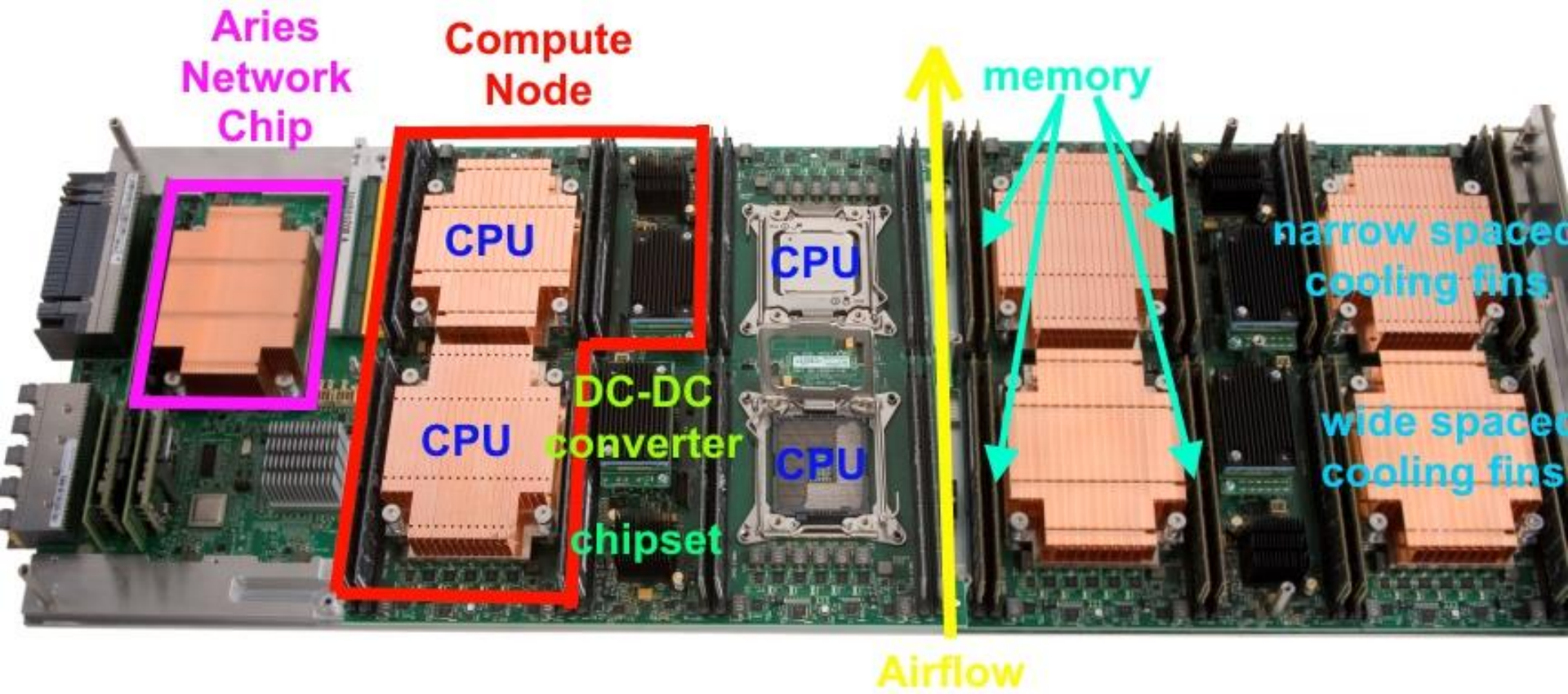
Last In, First Out (LIFO)

Priority Queue



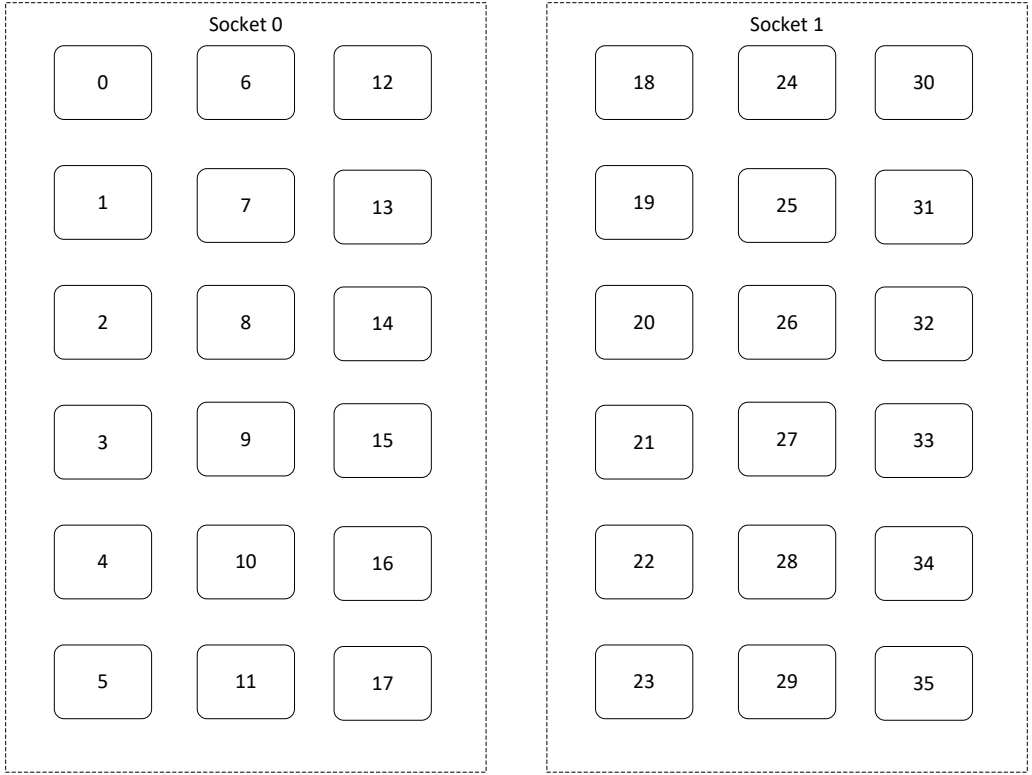
Generate Job Script

```
#!/bin/bash -l
#SBATCH --account=uzh8
#SBATCH --job-name=hpc_test
#SBATCH --time=00:30:00
#SBATCH --nodes=1
#SBATCH --ntasks-per-core=1
#SBATCH --ntasks-per-node=128
#SBATCH --cpus-per-task=1
#SBATCH --partition=normal
#SBATCH --constraint=mc

export OMP_NUM_THREADS=$SLURM_CPUS_PER_TASK

srun hostname
```

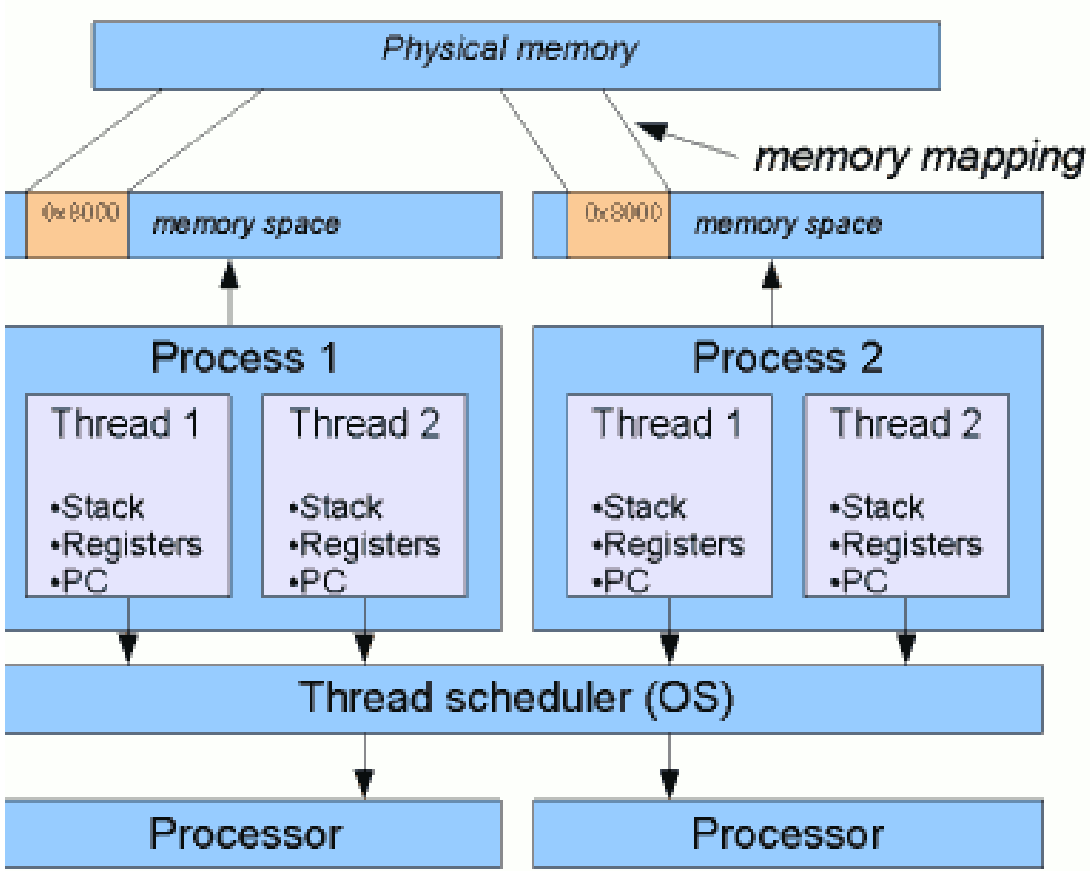


Cray XC40 Blade (daint-mc)

Processor Cores Layout



Processes & Threads

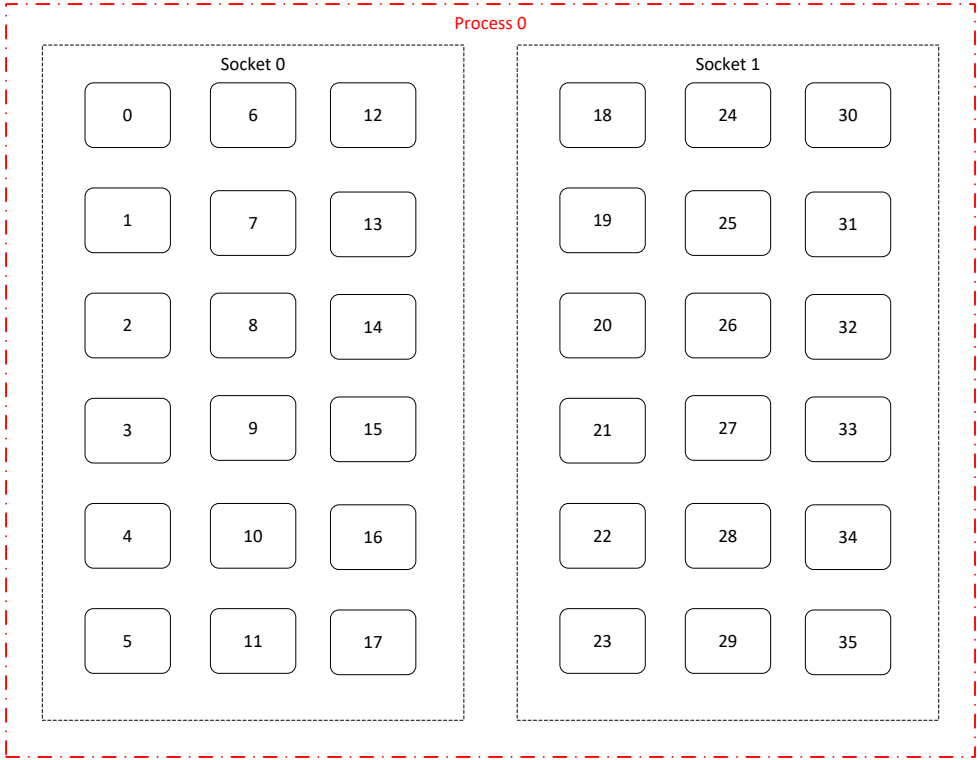


MPI Layout

- --ntasks-per-core=1
- --ntasks-per-node=36
- --cpus-per-task=1

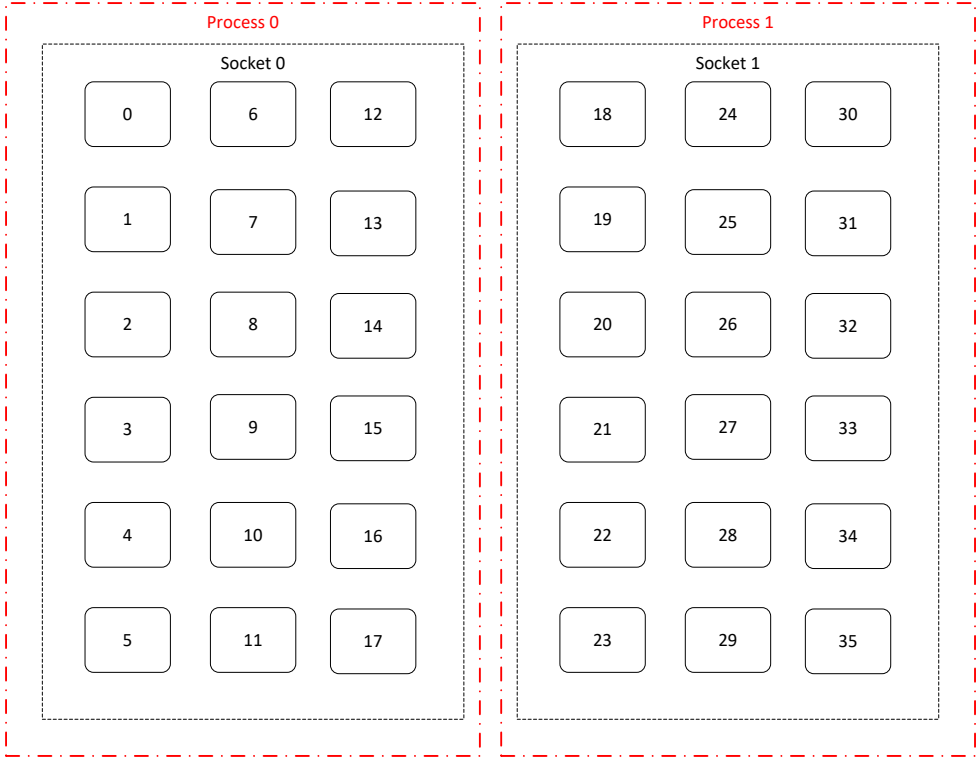
OpenMP Threads Layout

- --ntasks-per-core=1
- --ntasks-per-node=1
- --cpus-per-task=36



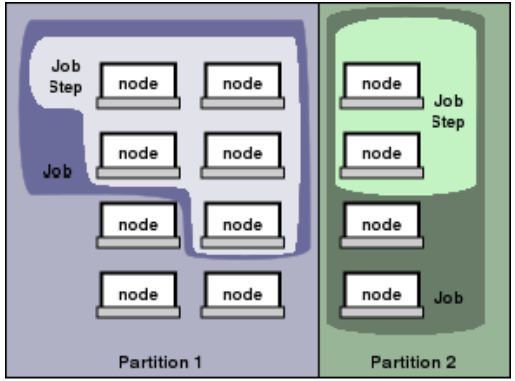
Hybrid Layout

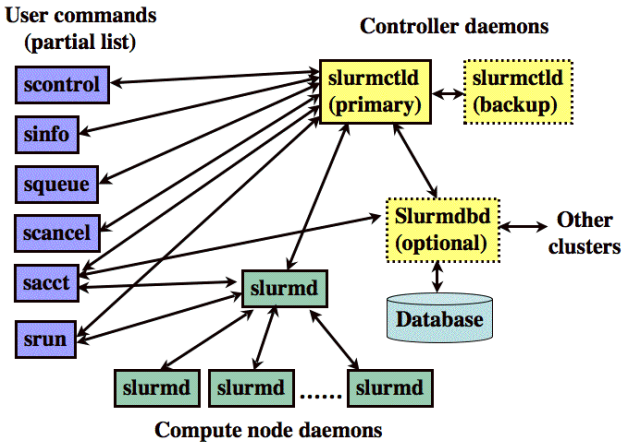
- --ntasks-per-core=1
- --ntasks-per-node=2
- --cpus-per-task=18



SLURM Job Submission

- ”sbatch” to submit a job
- “squeue” to see jobs
  - Pending jobs: -t pending
  - Running jobs: -t running
  - Your jobs: -u username
- “sinfo” to see partition information
  - Normal partition: -p normal
  - Debug partition: -p debug
- “srun” to run a job step
  - Always use “srun” in scripts
  - You can use it for interactive sessions
    - Not recommended if avoidable
- “scancel” to cancel a job
- “accounting” for usage
  - CHF 0.376 per “node hour”

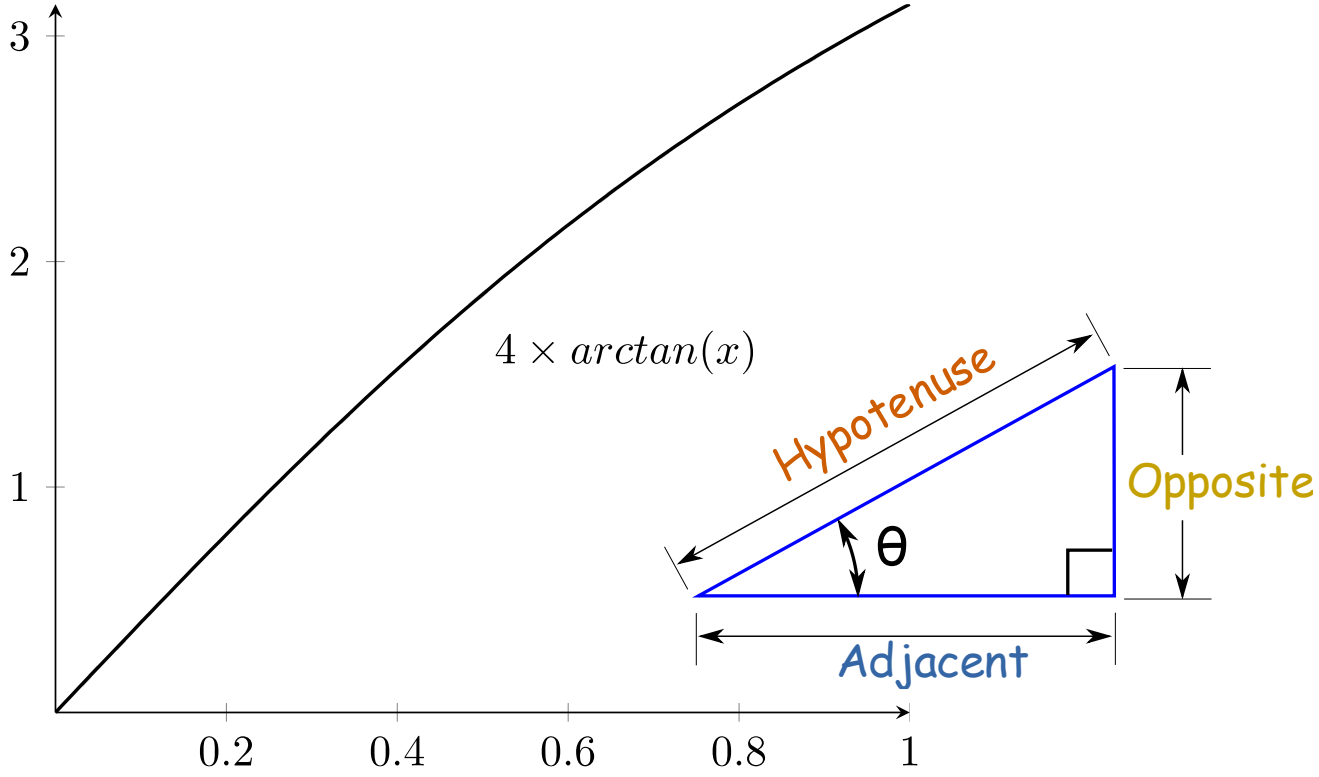




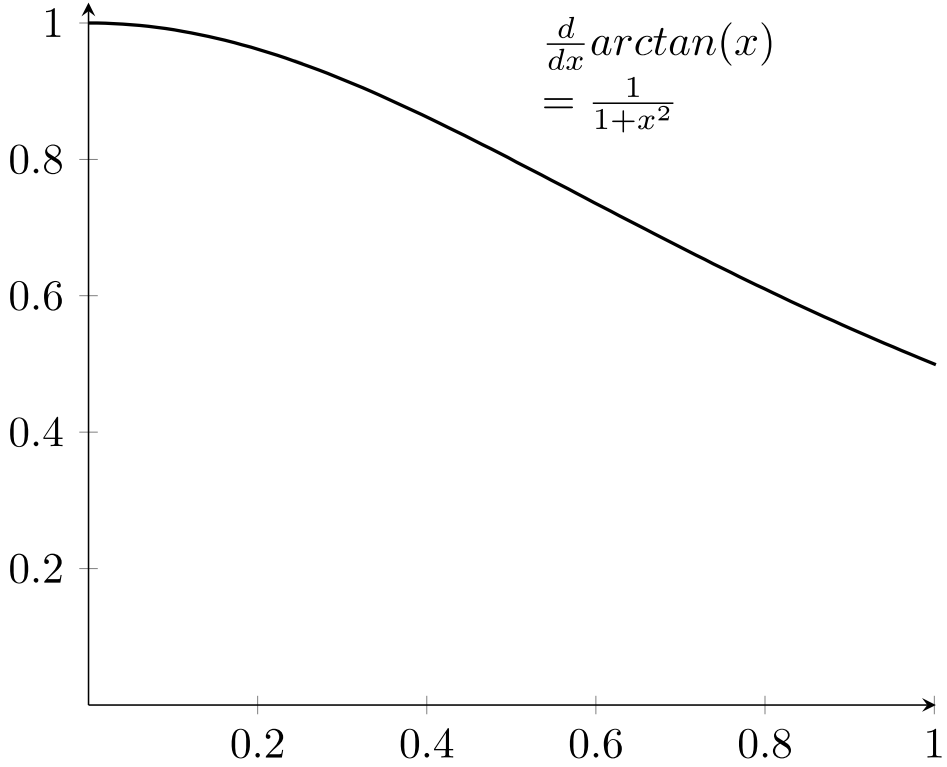
Sample Program

Calculate pi

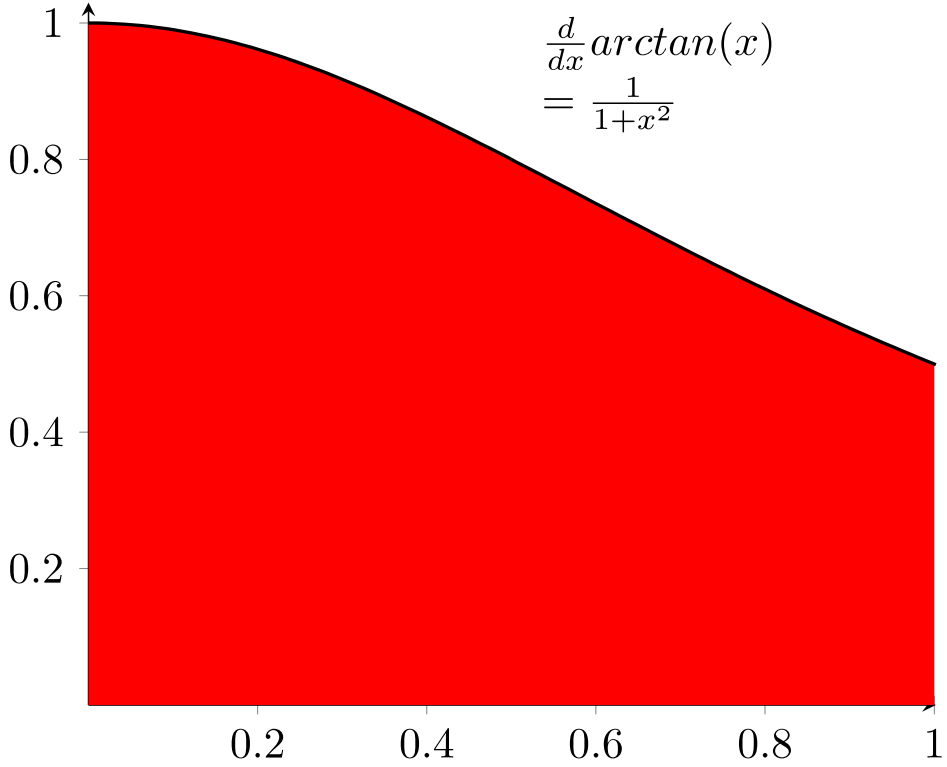
Calculate pi using arctan



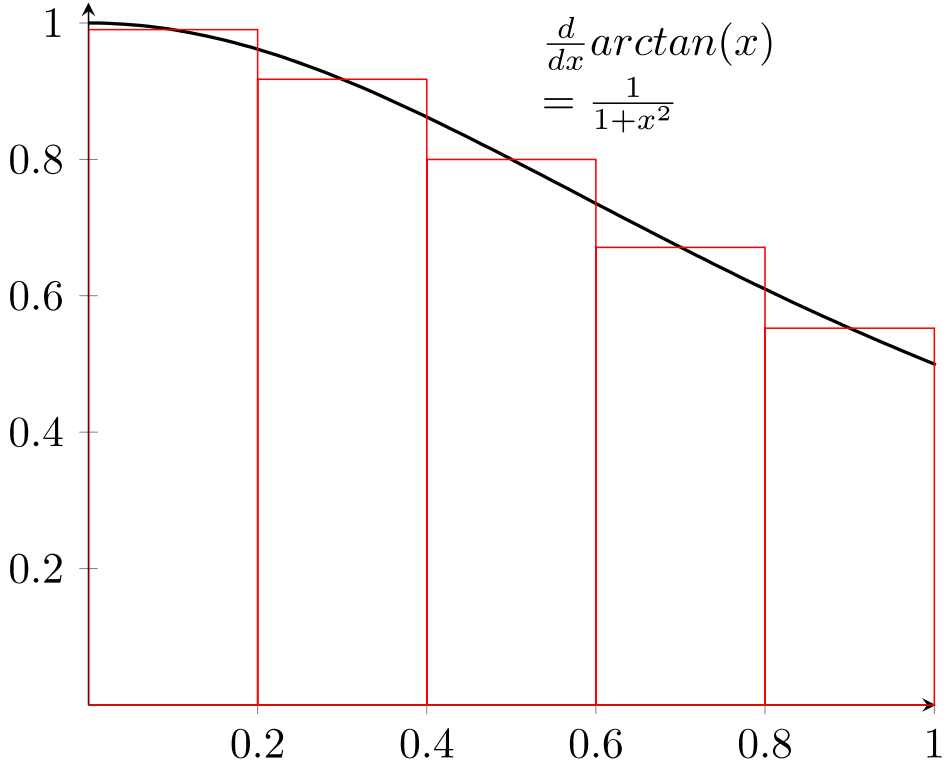
Derivative is well known



Integrate to get $\frac{\pi}{4}$

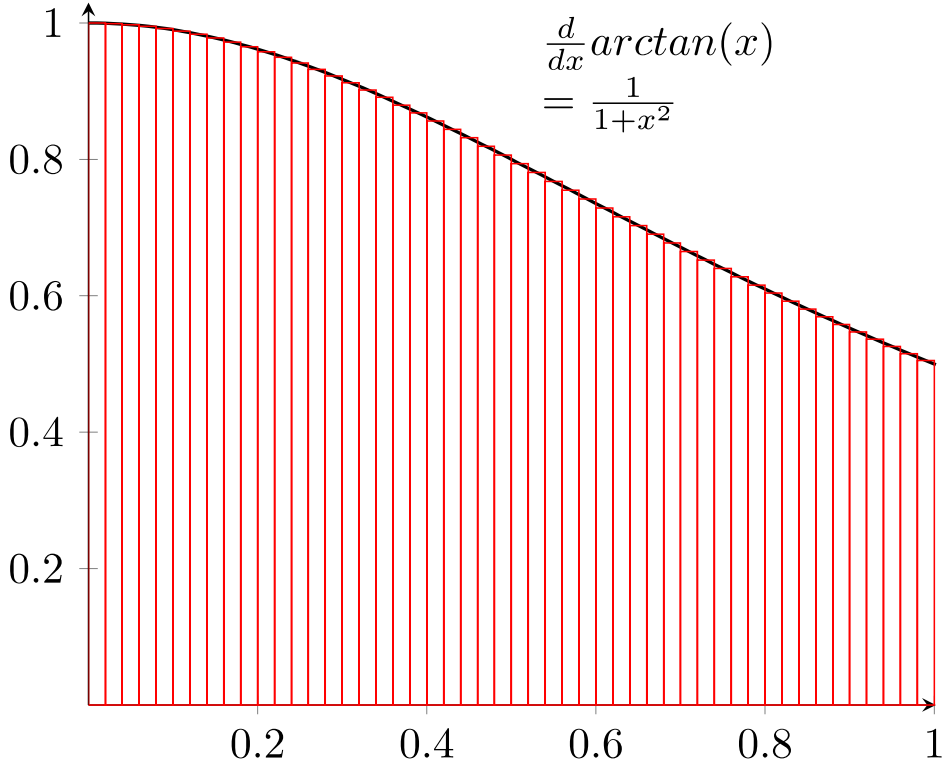


Integrate numerically



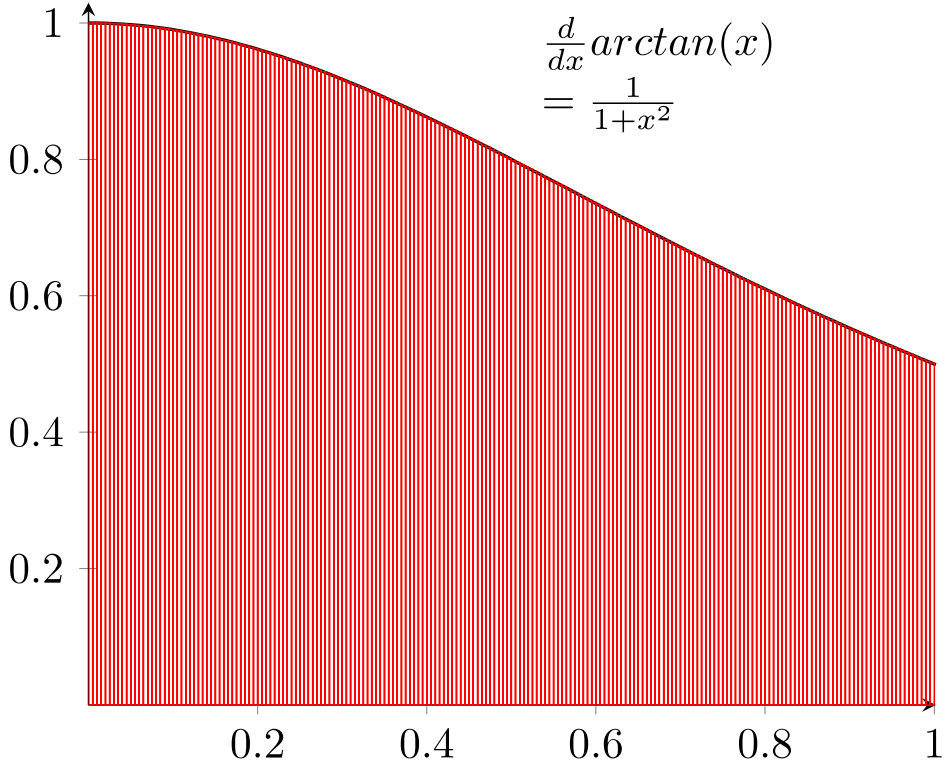
= 3.144925

Improve Accuracy



= 3.141625

We haven't even started!



= 3.141595

Integration in Python

```
N=5
dx=1.0/N
s = 0
for x in range(0, N):
  s += dx * 1 / ( 1 + ((x+0.5)*dx)**2 )
print(s*4)

dhcp-94-191:cpi$ python integrate.py
3.1449258640033277
```

Timing.

Is this good?

Threads	Seconds
1	4.550
2	2.291
3	1.624
4	1.258
5	1.041
6	0.8988
7	0.7989
8	0.7217
9	0.6415
10	0.5774
11	0.5249
12	0.4811
13	0.4441
14	0.4124
15	0.3849
16	0.3609
17	0.3397
18	0.3208

Threads	Seconds
19	0.3053
20	0.2897
21	0.2768
22	0.2644
23	0.2528
24	0.2423
25	0.2326
26	0.2237
27	0.2154
28	0.2077
29	0.2006
30	0.1939
31	0.1876
32	0.1818
33	0.1763
34	0.1711
35	0.1662
36	0.1616

Timing Plot

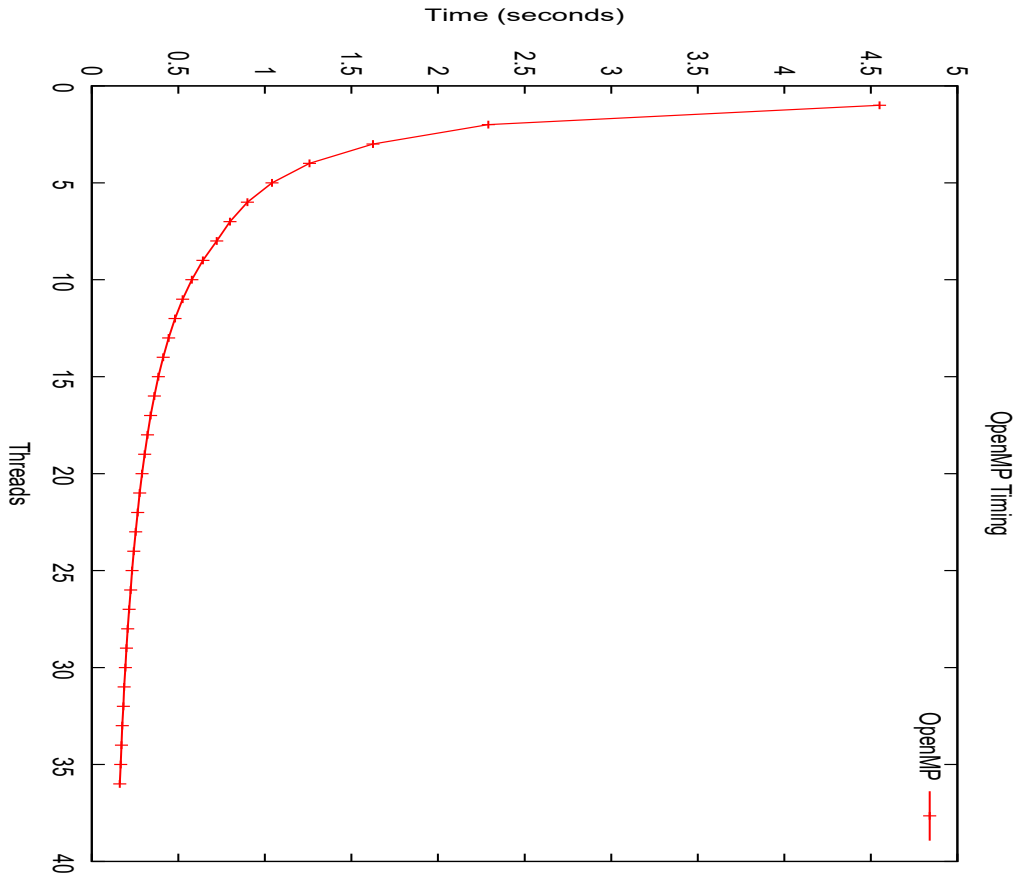
set title 'OpenMP Timing'

set xlabel 'Threads’

det ylabel 'Time (seconds)'

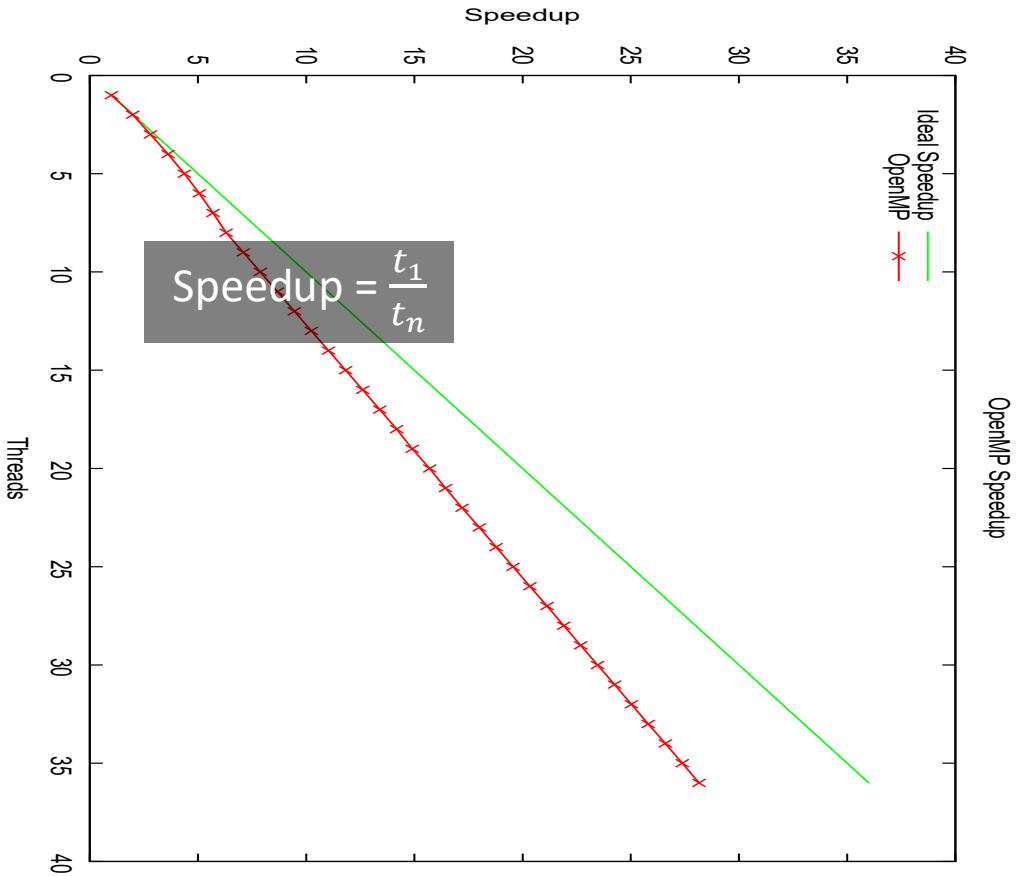
set key top right

plot "cpi_openmp.dat" u 1:2 w lp lw 2 t 'OpenMP'



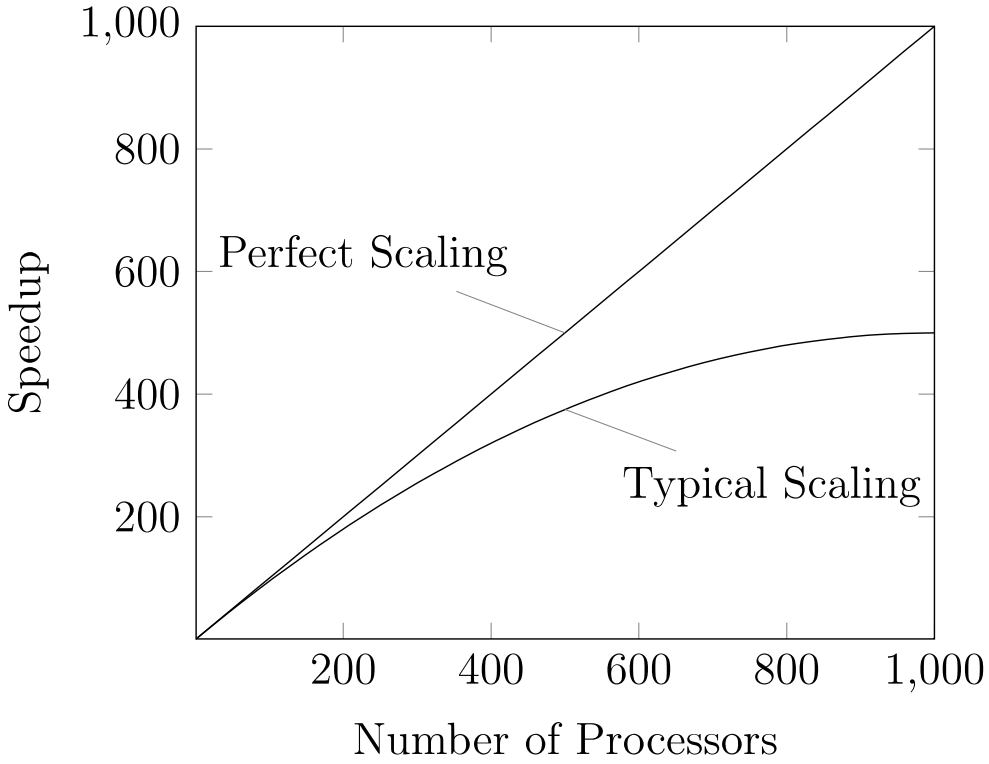
“Speedup” Plot

```
set title 'OpenMP Speedup'
set xlabel 'Threads'
set ylabel 'Speedup'
set key top left
plot x lc 2 lw 2 t 'Ideal Speedup’,\
   "cpi_openmp.dat" u 1:(4.55/$2)\
   w lp lc 1 lw 2 t 'OpenMP'
```

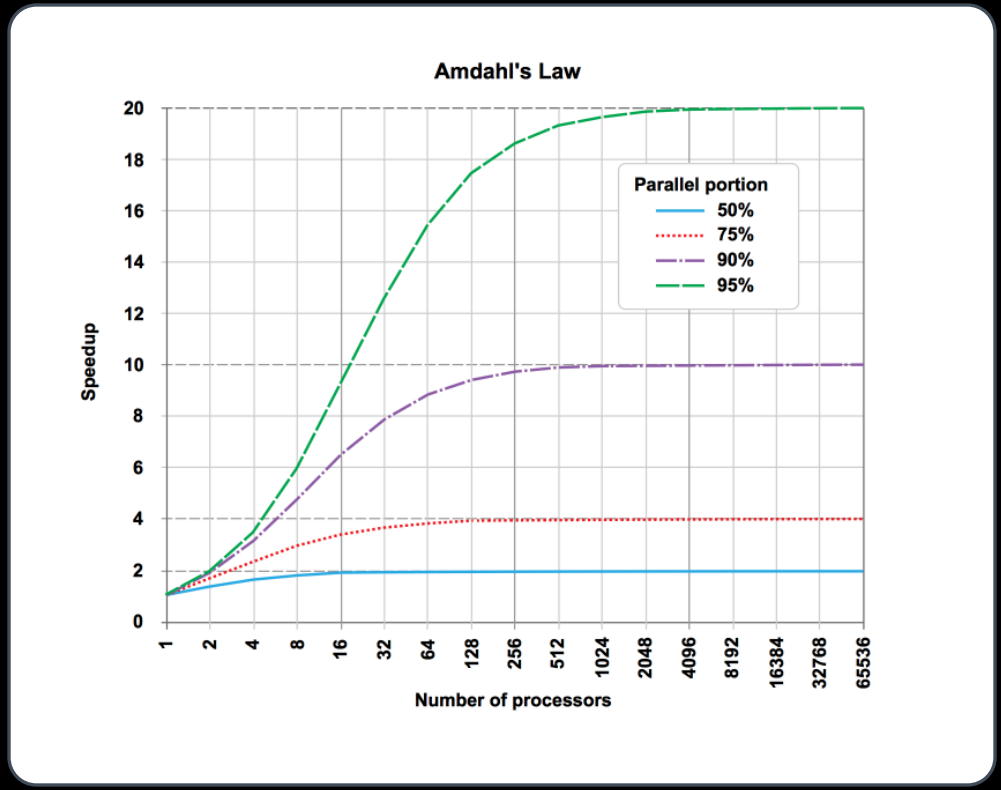


$$\text{Speedup} = \frac{t_1}{t_n}$$

How does it scale?



Why won’t it scale?



Makefile (for the C versions)

```
all:   cpi_openmp cpi_mpi

cpi_openmp:  cpi_openmp.c
       cc -g -O3 -o cpi_openmp -fopenmp cpi_openmp.c

cpi_mpi:     cpi_mpi.c
       cc -g -O3 -o cpi_mpi cpi_mpi.c

clean:
       rm -f cpi_openmp cpi_mpi
```

Target

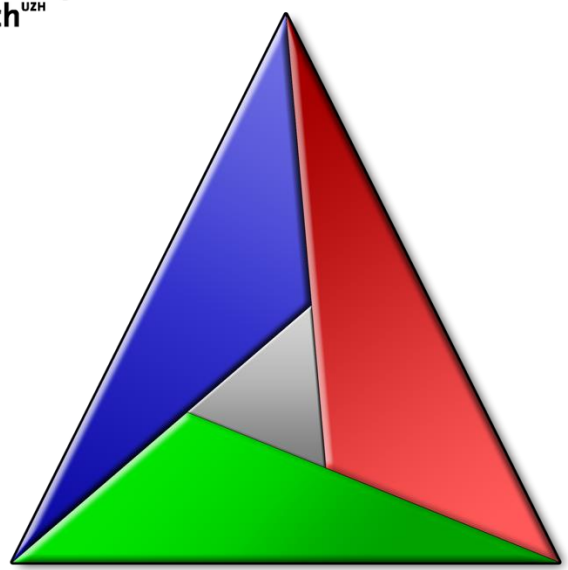
Dependencies

Triggered when newer

Build Rule(s)

Common “clean” target

University of Zurich[UZH]



Advanced Make Systems

General Recipe

```
SRC=$HOME/source
DST=$HOME/install



cd $SRC
./configure\
  --prefix=$DST
make
make install
```

```
SRC=$HOME/source
DST=$HOME/install
mkdir $HOME/build
cd $HOME/build
cmake\
 -DCMAKE_INSTALL_PREFIX:PATH=$DST\
 $SRC
make
make install
```