# PARSOLPS

Tamás Fehér[1], Lorenz Hüdepohl[2]

[1]High Level Support Team, IPP

[2]Rechenzentrum Garching

WPCD SOLPS Working Session, 10-12.12.2014

# PARSOLPS project

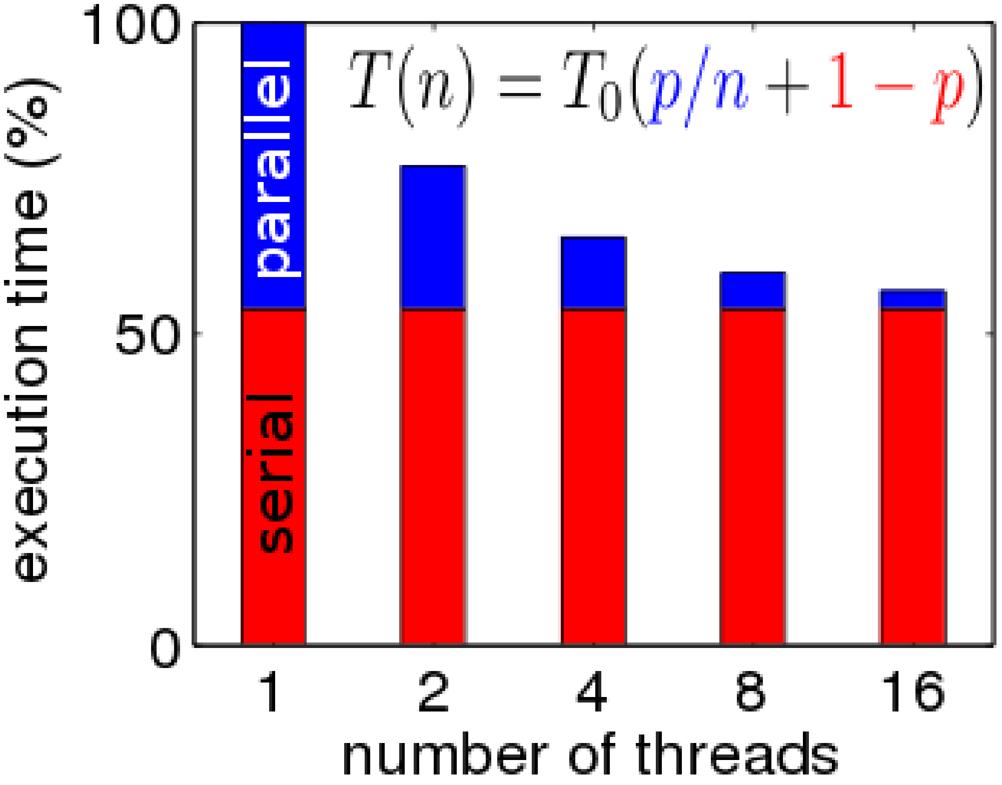
- Scrape-Off Layer Plasma Simulation
- Main components:
  - B2: fluid code for edge plasmas
  - EIRENE: Monte-Carlo code for neutral transport
- Improve parallel performance of SOLPS 5.0
  - OpenMP parallelization of B2.5
  - Couple with EIRENE

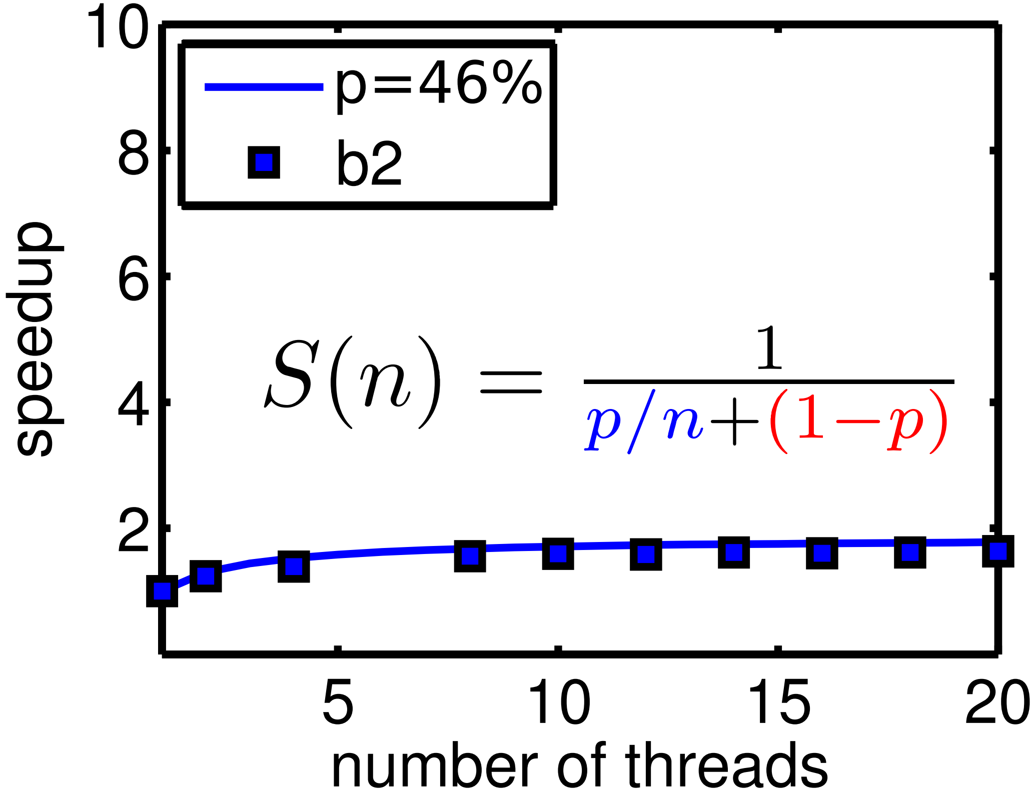
# Outline

- Quick overview:
  - Previous work
  - Improvements & speedup
- Details:
  - OpenMP overhead
  - Speedup of B2 Subroutines
  - Bottlenecks
  - Correctness

# Previous work & Amdahl's law

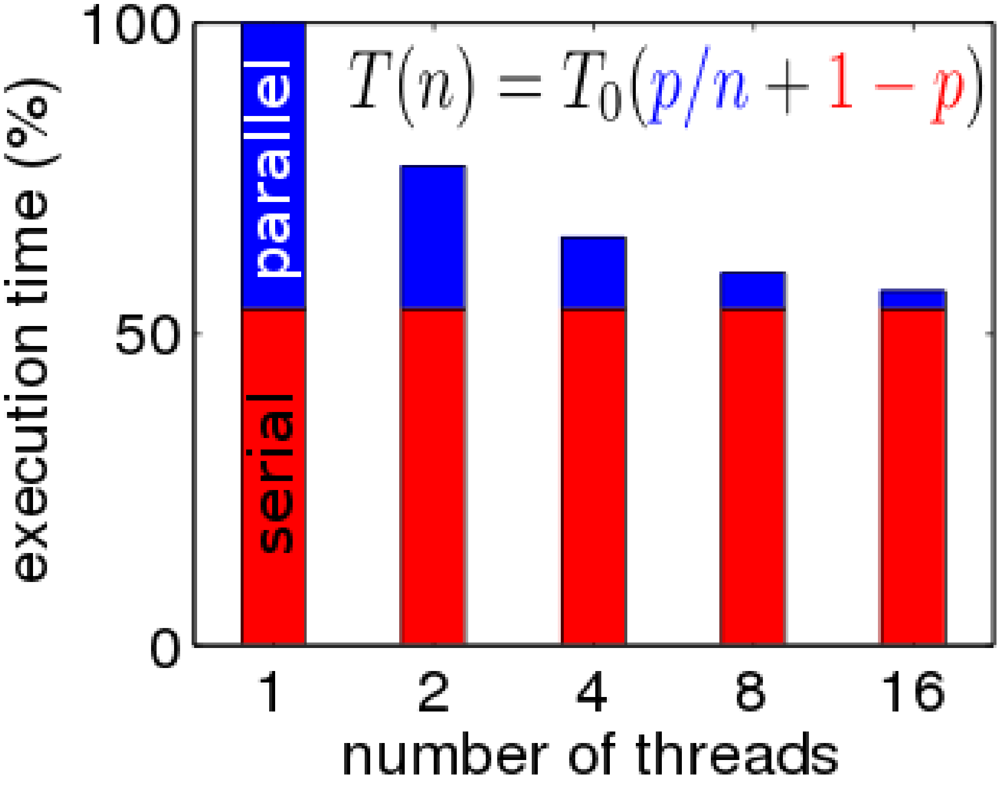
- 4 OpenMP parallel loops by F. Reid
- Speedup is limited by the parallel fraction
- Testcase: ITER H+T+He+Be+Ne+W: 98 species

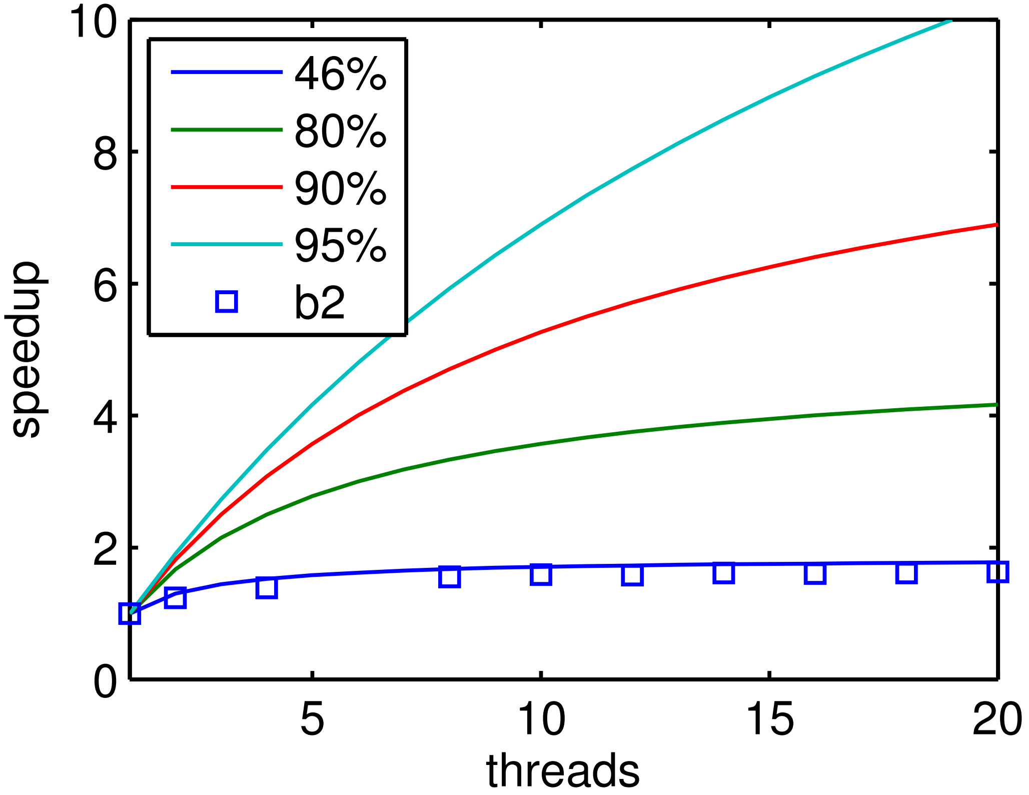




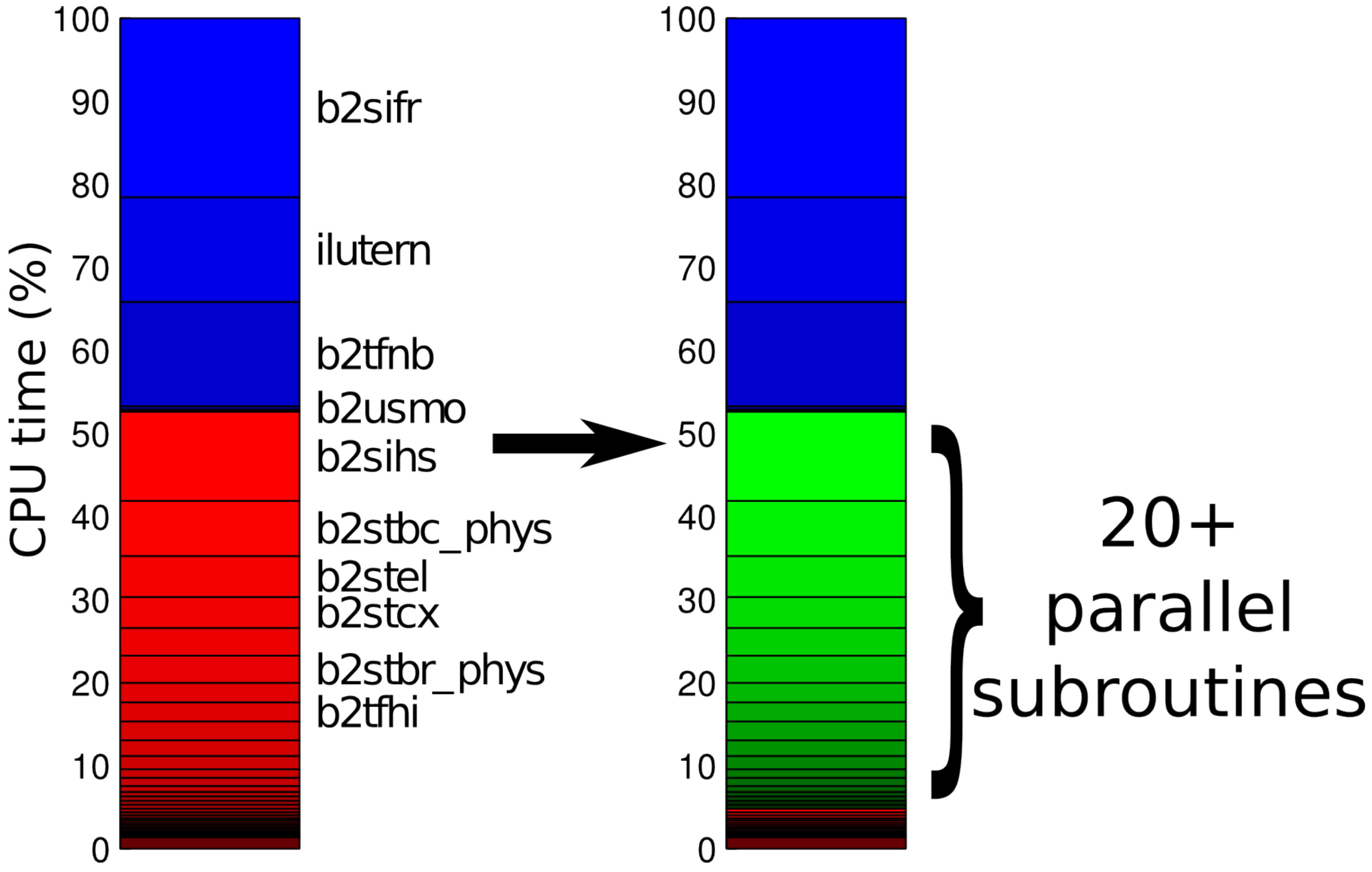
# Previous work & Amdahl's law

- 4 OpenMP parallel loops by F. Reid
- Speedup is limited by the parallel fraction
- Testcase: ITER H+T+He+Be+Ne+W: 98 species



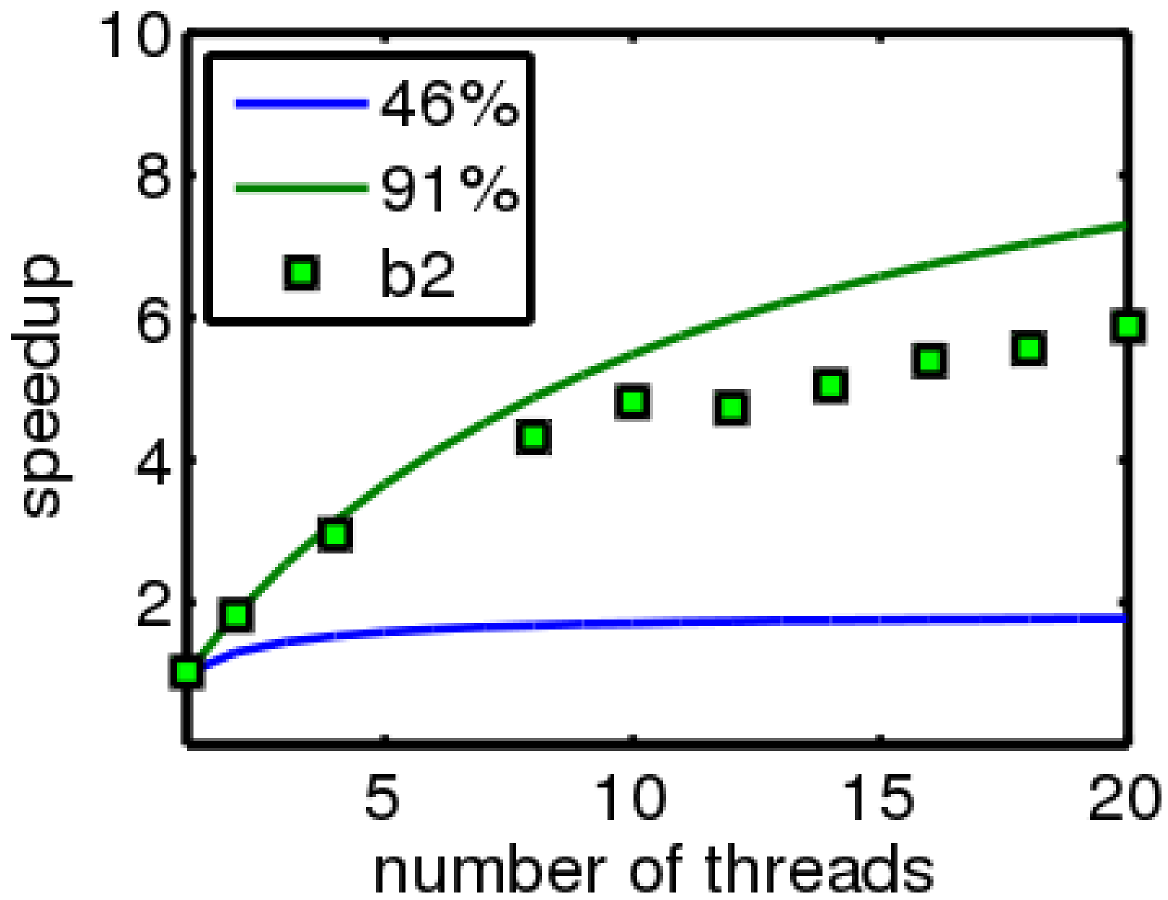


# Improving parallelization



# B2 speedup: 6x

- Testcase: ITER D+T+He+Be+Ne+W
- Parallel fraction 91%
- Run on 1 Ivy-Bridge node (2x10 cores) @ RZG Hydra



# Outline

Quick overview:

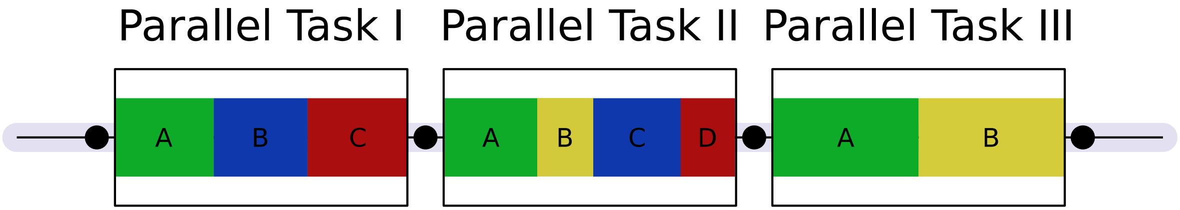
  Previous work

  Improvements & speedup

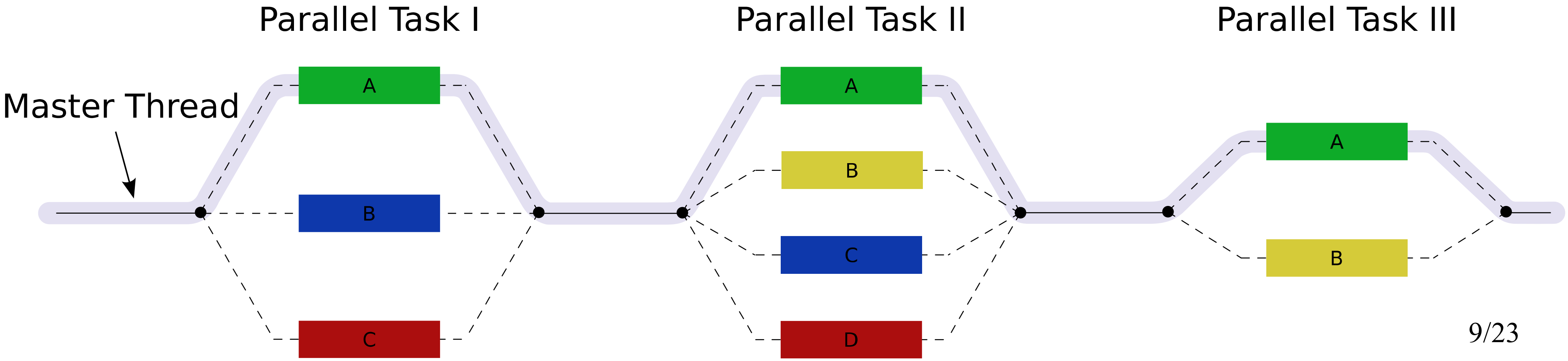
- Details:
  - OpenMP overhead
  - Speedup of B2 Subroutines
  - Bottlenecks
  - Correctness

# OpenMP parallelization

## Sequential execution:



## Parallel execution:



# OMP loop overhead: 1-4 µs

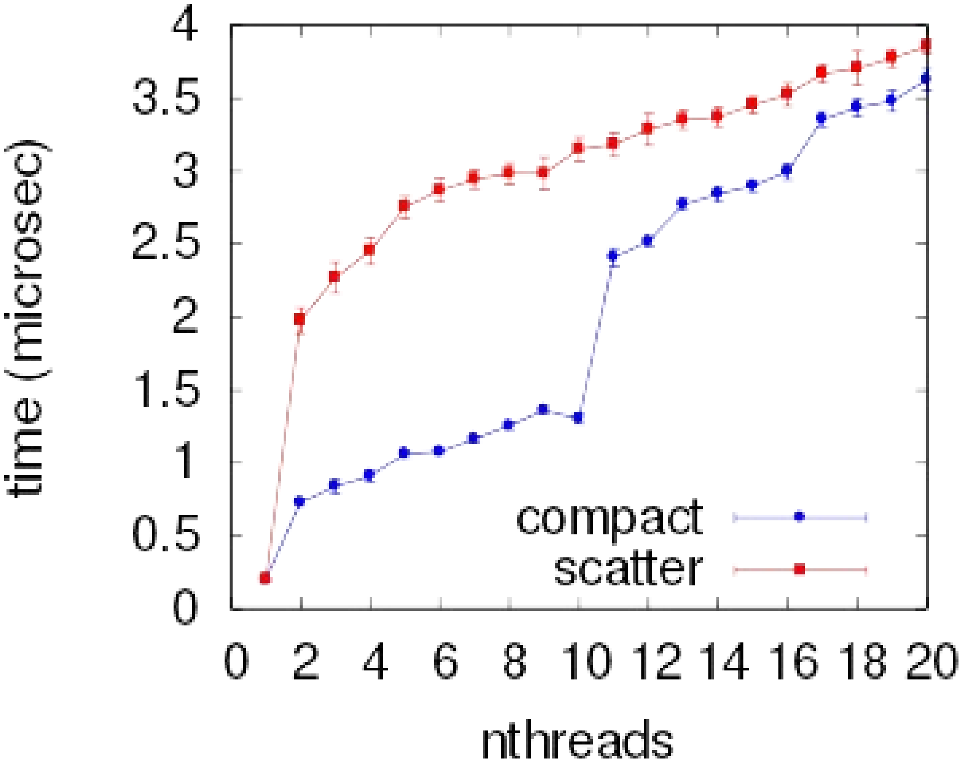
## EPCC OpenMP Microbenchmarks

Sequential code

```
for ( j = 0 ; j < innerreps ; j ++) {
   delay ( delaylength ) ;
}
```

Parallel code

```
for ( j = 0 ; j < innerreps ; j ++) {
#pragma omp parallel for
   for ( i = 0 ; i < nthreads ; i ++) {
      delay ( delaylength ) ;
   }
}
```



# OMP loop overhead: 1-4 μs

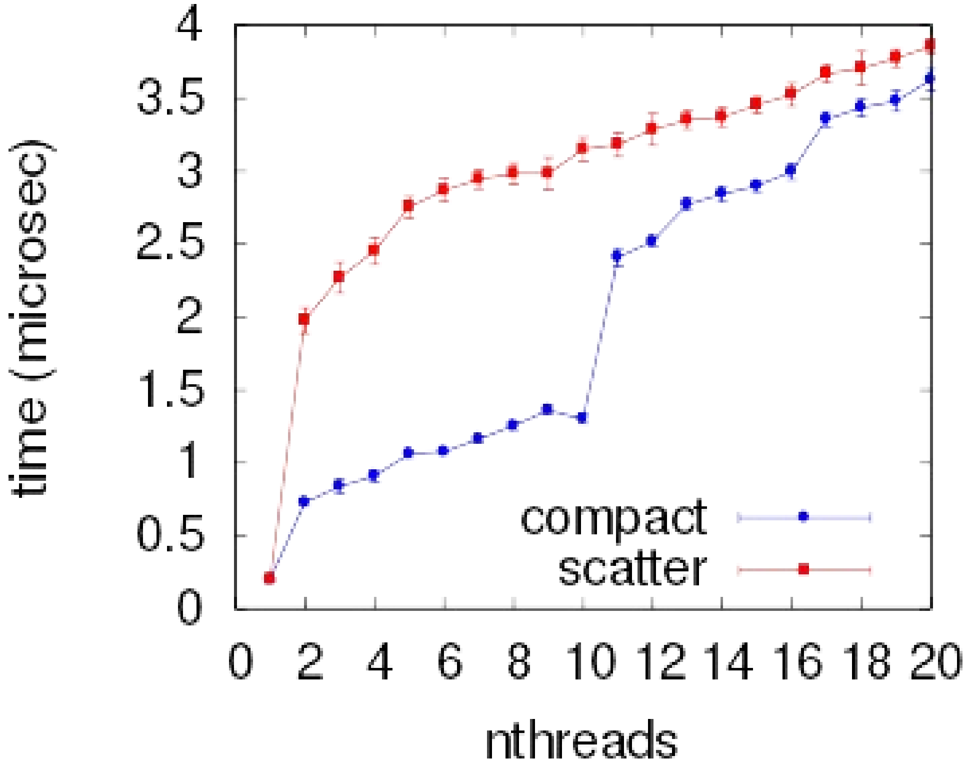
## EPCC OpenMP Microbenchmarks

Sequential code

```
for ( j = 0 ; j < innerreps ; j ++) {
   delay ( delaylength ) ;
}
```

Parallel code

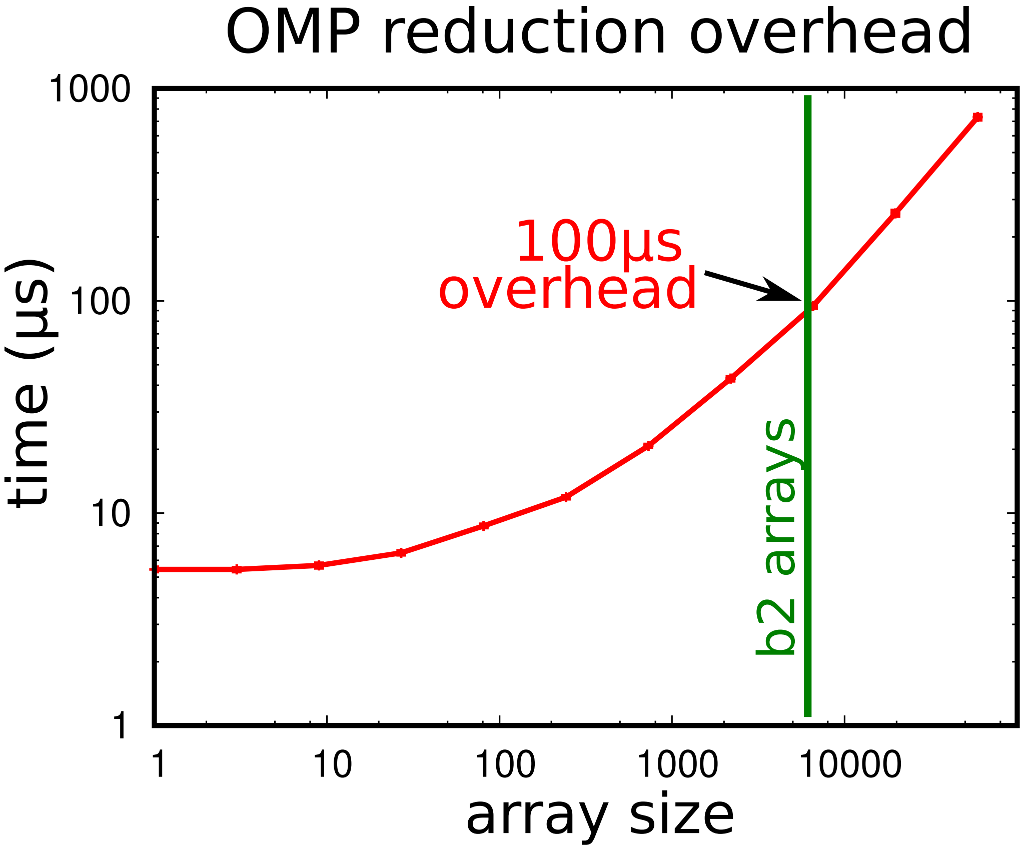
```
for ( j = 0 ; j < innerreps ; j ++) {
#pragma omp parallel for
   for ( i = 0 ; i < nthreads ; i ++) {
      delay ( delaylength ) ;
   }
}
```



- Memory transfer of 1 radial grid (98x38 elements) 4μs ✗
- Arithmetics with several grids >20μs
- 3D arrays (species & radial grid) (98x98x38) ~ 1ms ✓

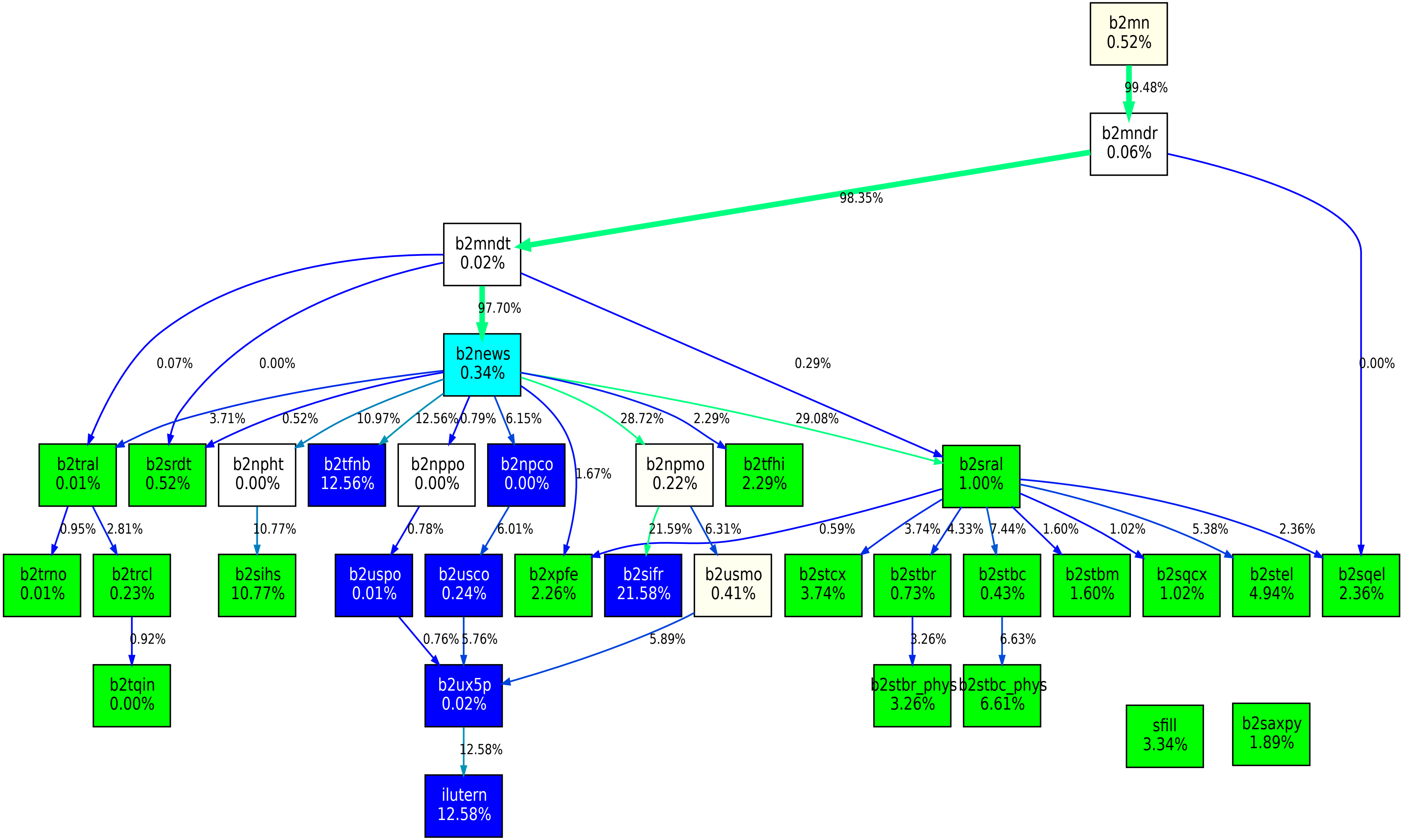
# OpenMP Reduction Overhead

```
!$omp parallel do private(is,ix,iy) &
!$omp reduction(+: s)
do is = 0, ns-1
  do iy = -1, ny
    do ix = -1, nx
      s(ix,iy) = s(ix,iy) + &
      a(ix,iy,is)* r(ix,iy,is) &
      * vol(ix,iy) * ne(ix,iy)
    enddo
  enddo
enddo
```

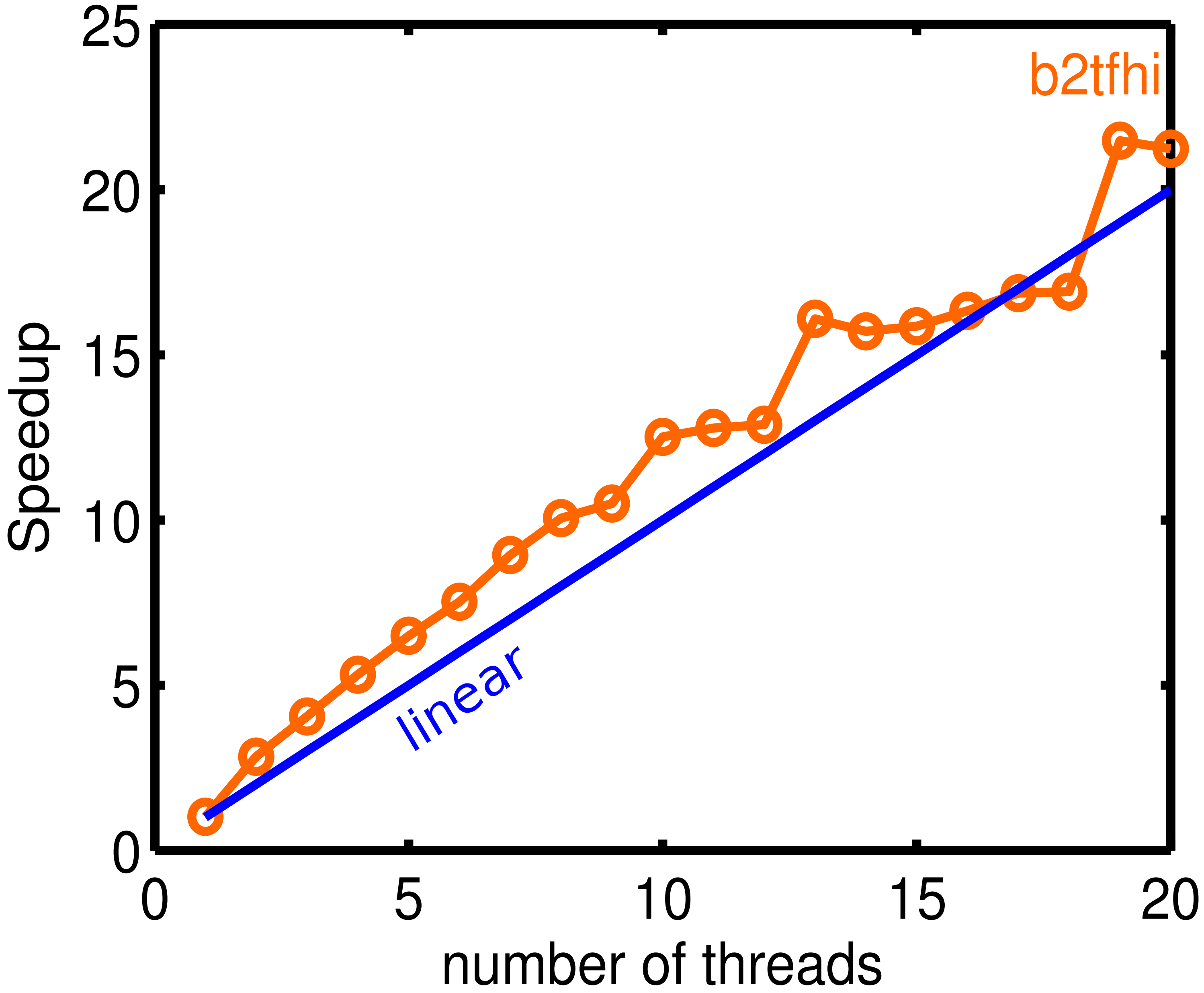


500 µs execution time

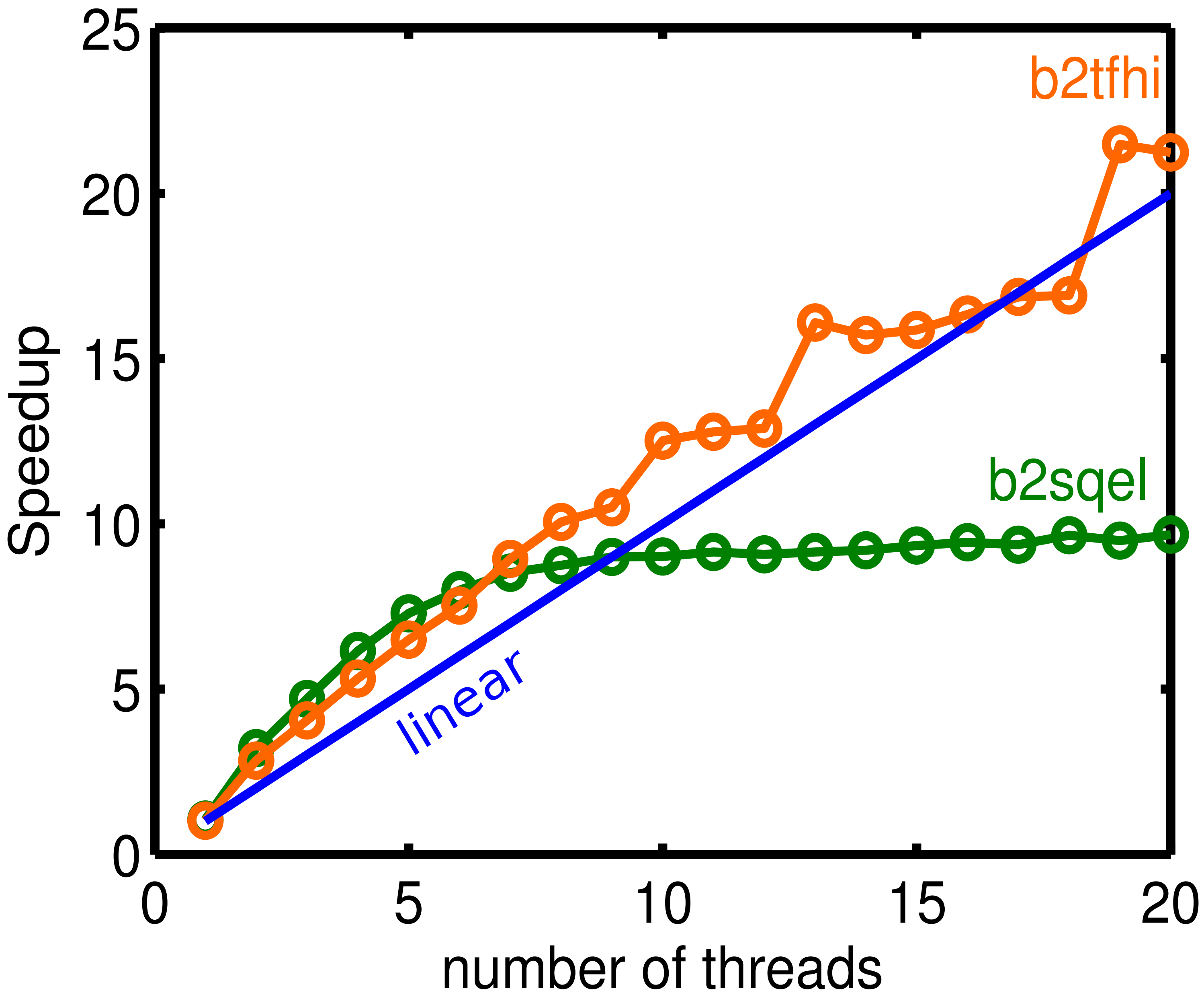
# B2 Callgraph



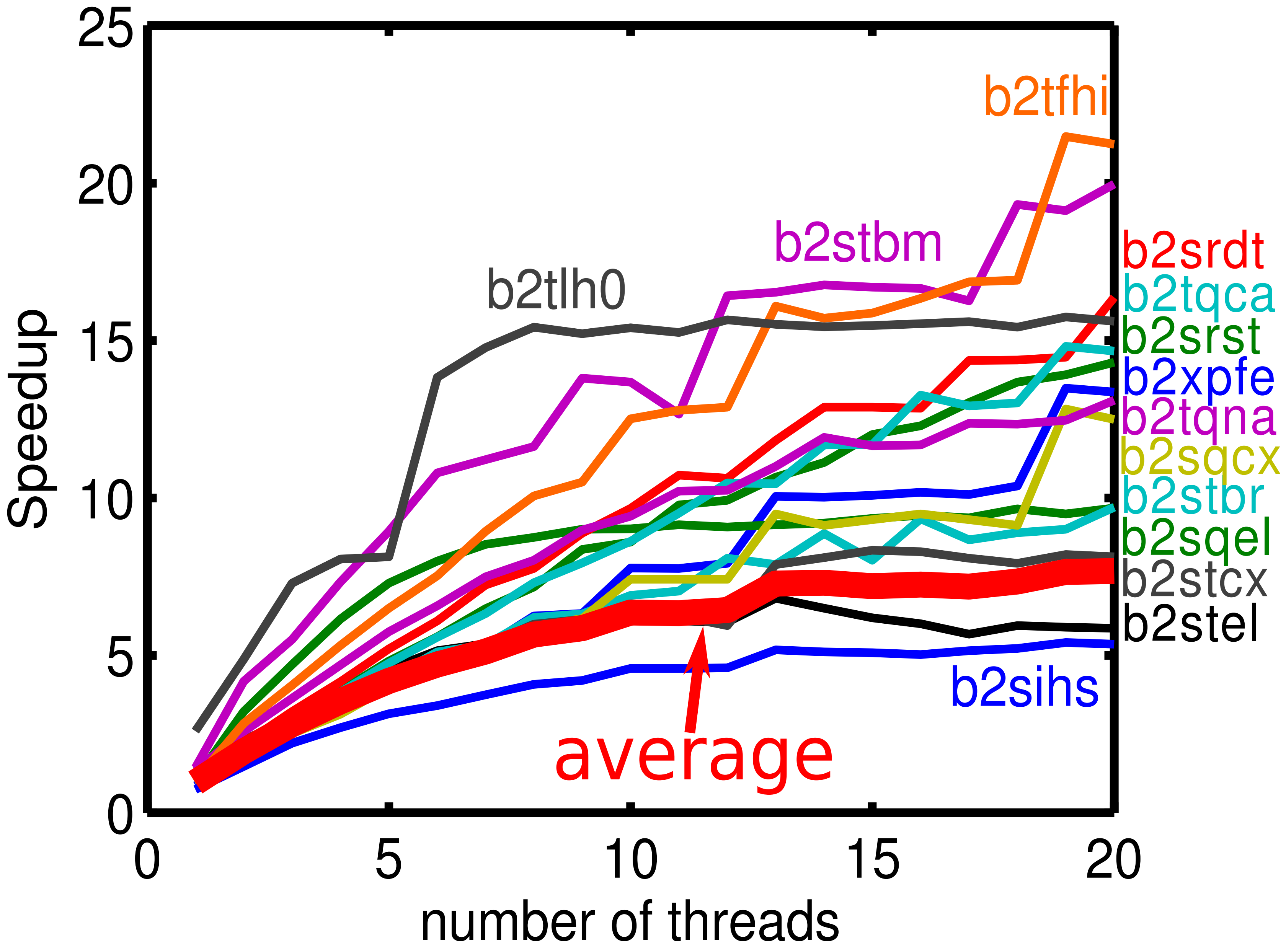
# Speedups


b2tfhi
linear
Speedup
number of threads
0
5
10
15
20
25
0
5
10
15
20

# Speedups


b2tfhi
b2sqel
linear
Speedup
number of threads
0
5
10
15
20
25

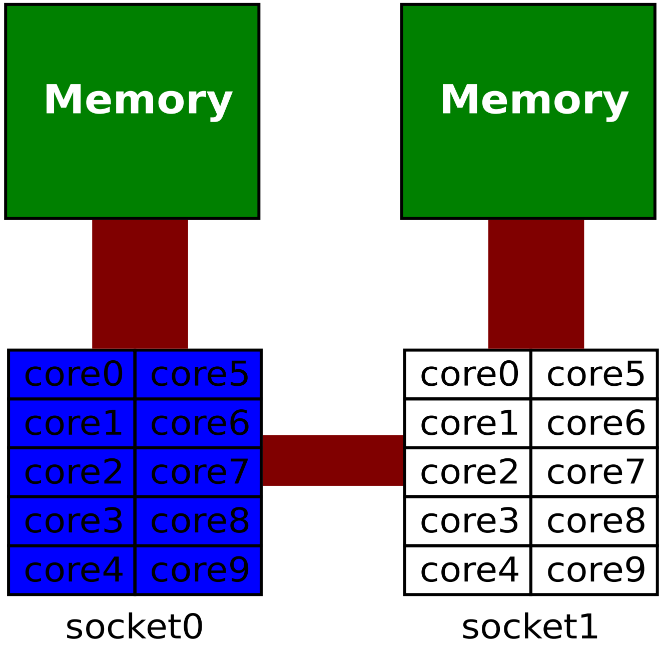
# Speedups

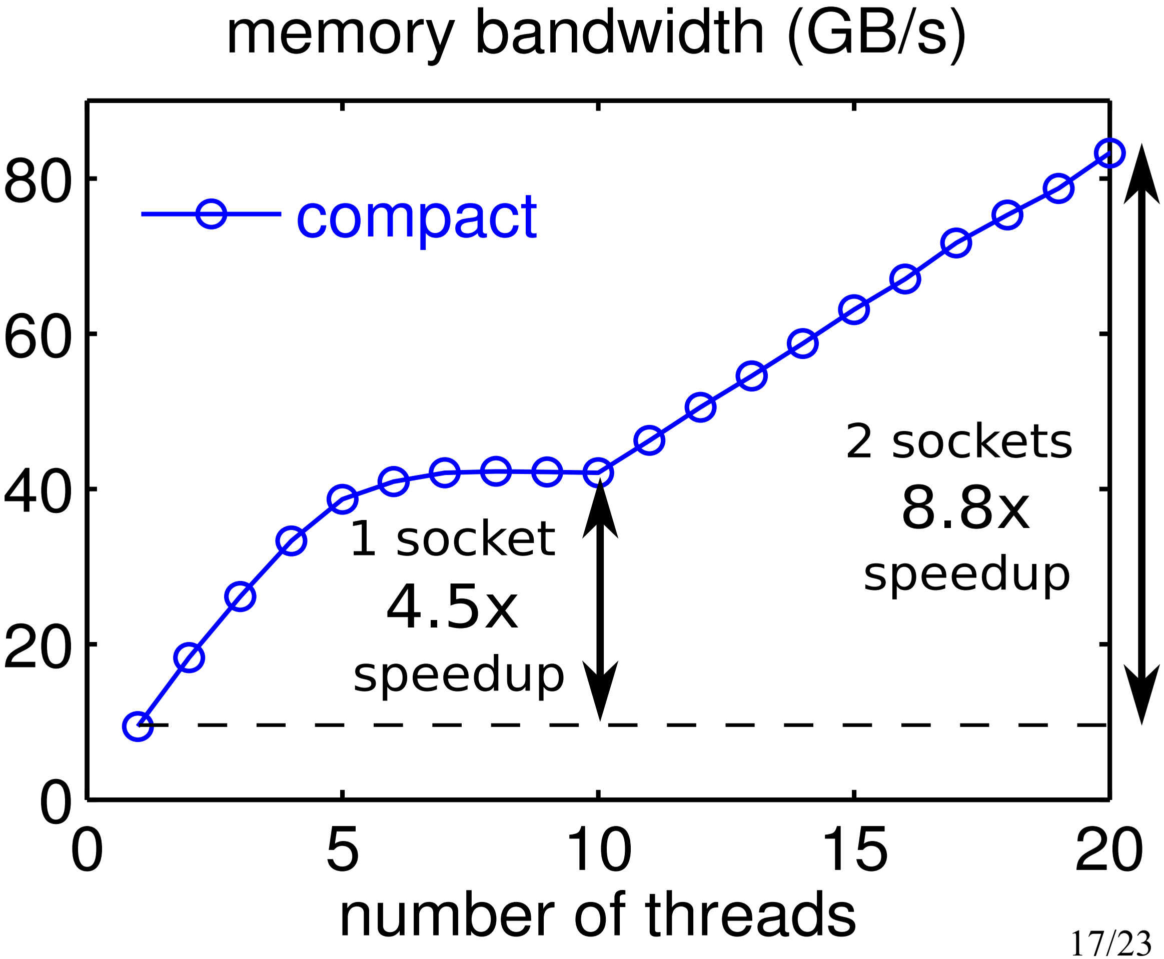


Speedup
number of threads
0
5
10
15
20
25
b2tfhi
b2stbm
b2tlh0
b2srdt
b2tqca
b2srst
b2xpfe
b2tqna
b2sqcx
b2stbr
b2sqel
b2stcx
b2stel
b2sihs
average

# Memory bottleneck

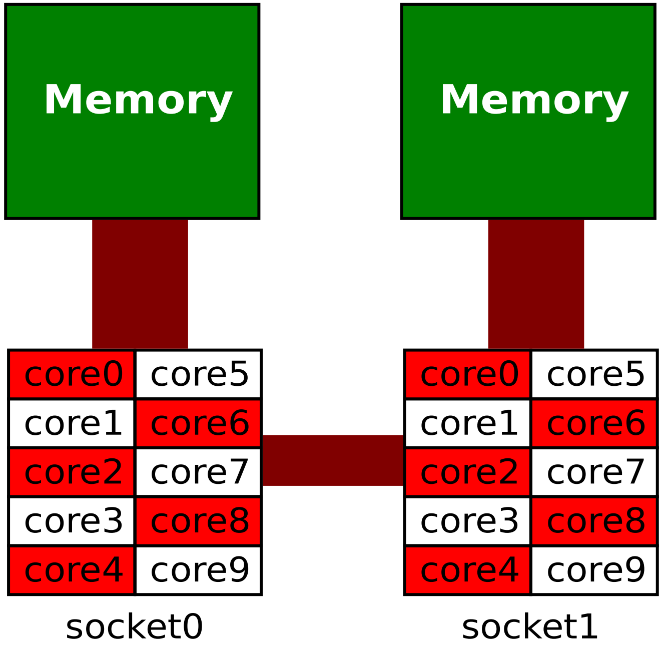
- Lot of data, few FLOPs / data
- Memory bandwidth limits the achievable speedup



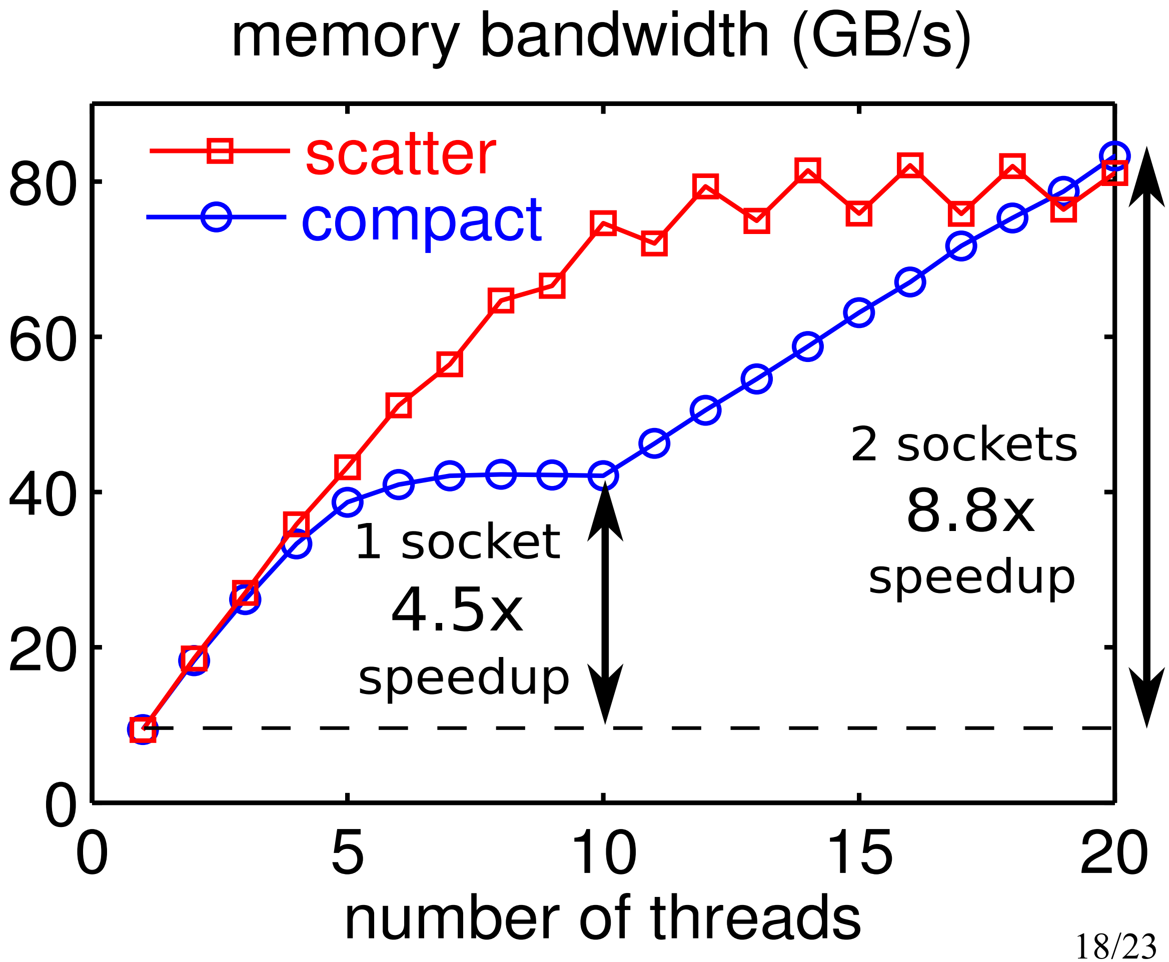


# Memory bottleneck

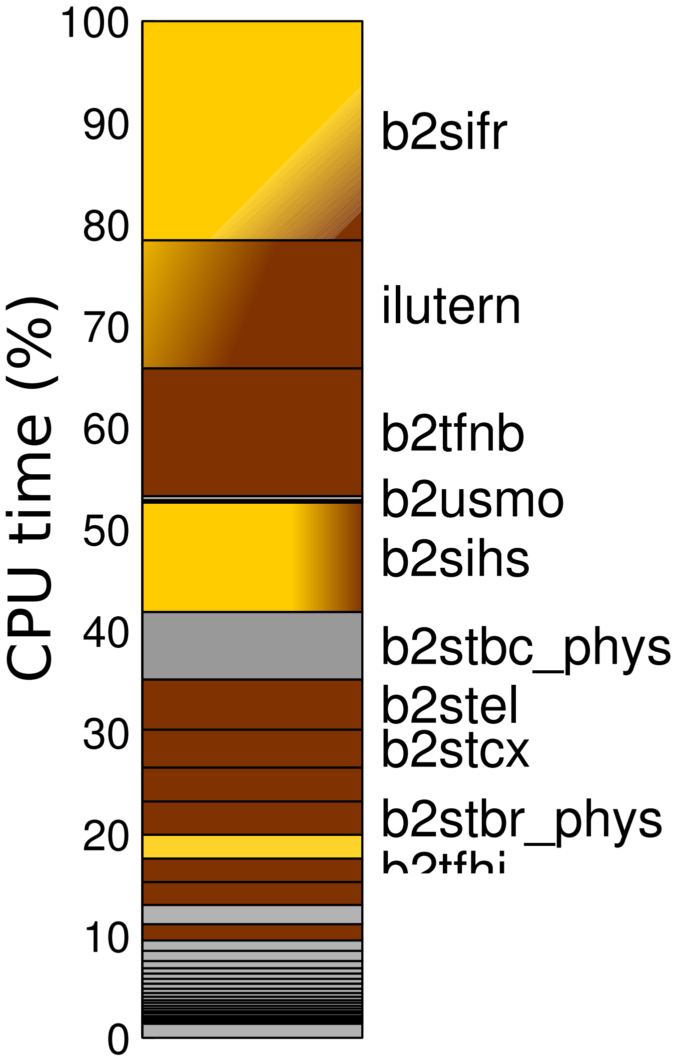
- Lot of data, few FLOPs / data
- Memory bandwidth limits the achievable speedup

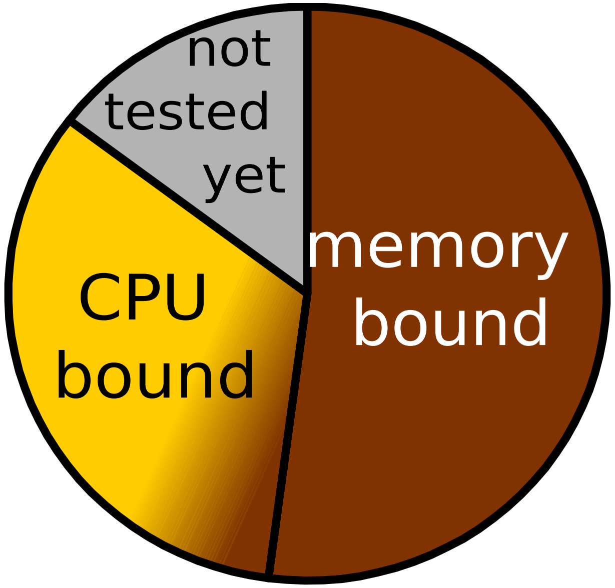


| socket0 | | socket1 | |
|---|---|---|---|
| core0 | core5 | core0 | core5 |
| core1 | core6 | core1 | core6 |
| core2 | core7 | core2 | core7 |
| core3 | core8 | core3 | core8 |
| core4 | core9 | core4 | core9 |

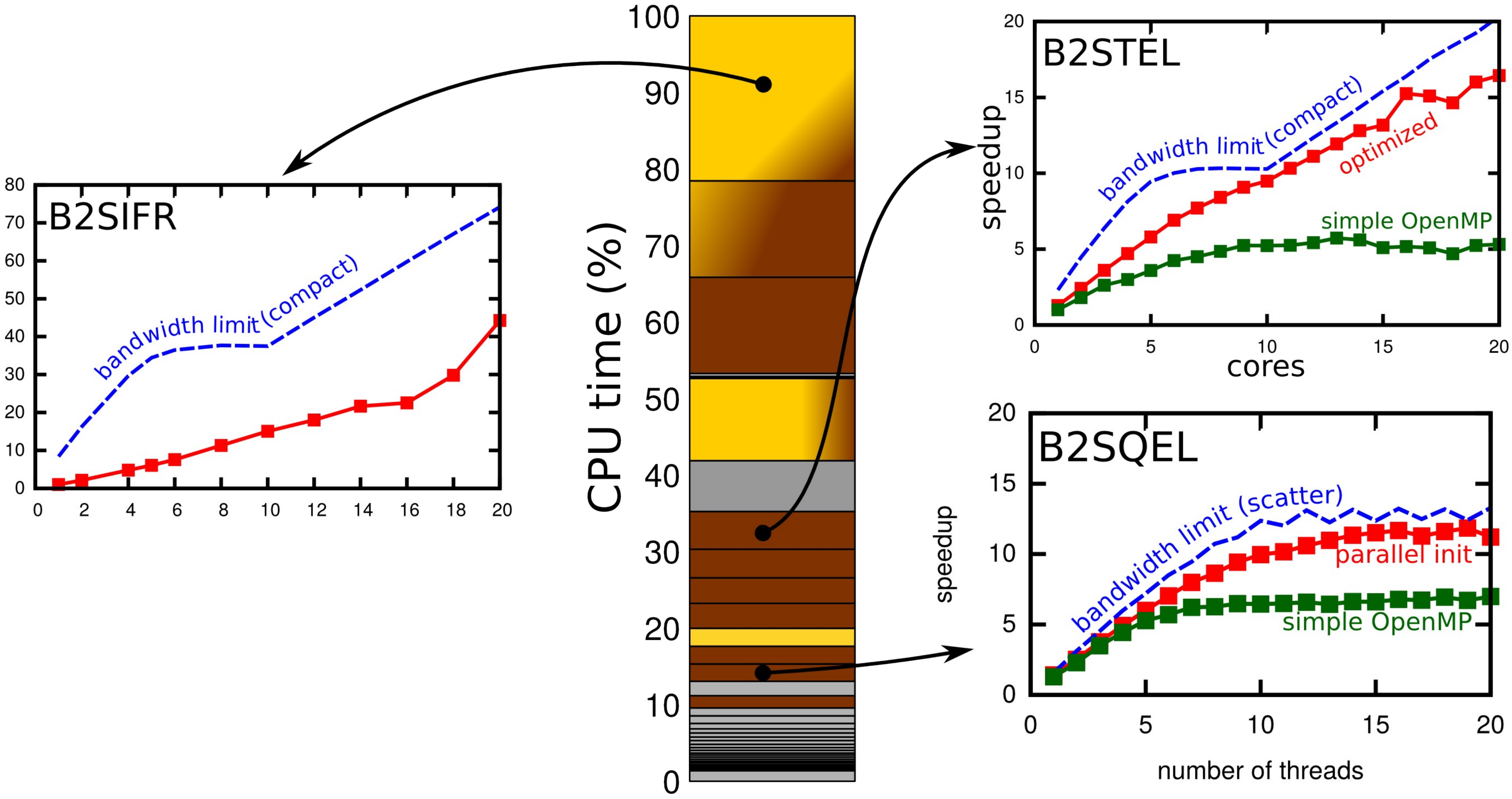


# Bottlenecks in B2





# Bottlenecks in B2



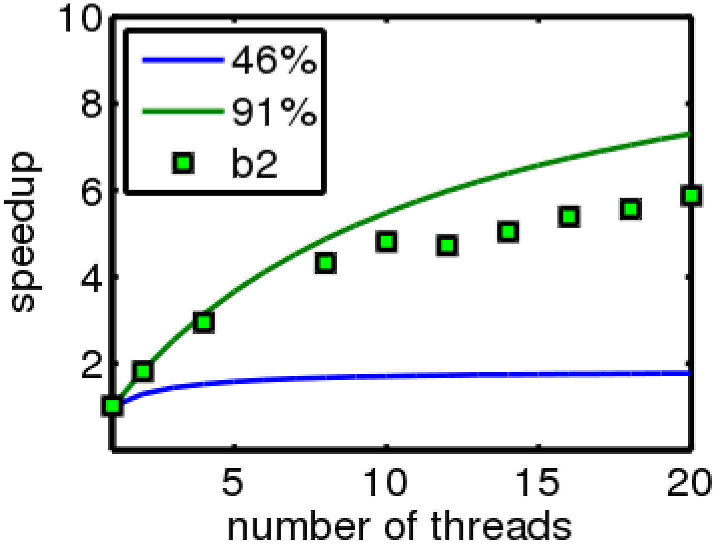
# Correctness

- Unit test framework:
  - Test programs generated
  - For every modified subroutine
  - Test all input/output data and side effects
- Complete tests:
  - Test the whole program
  - Bit-identical results

    (until a certain point in optimization)
- Tests are successful: relative error is $10^{-13}$

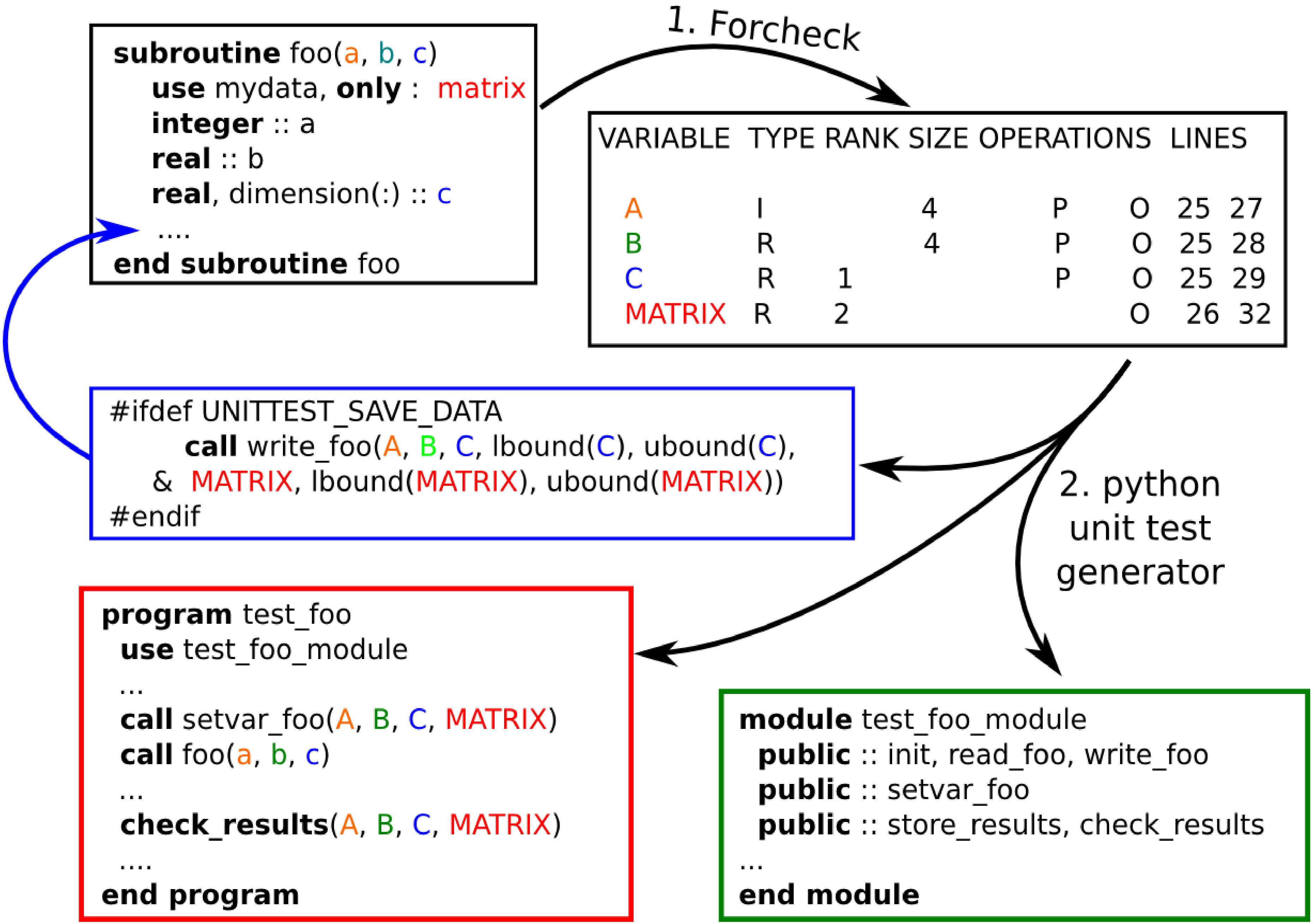
# Summary

- OpenMP parallelization of B2.5:
  - 20+ subroutines parallelized
  - 6x speedup for ITER test case
  - Carefully tested



- Ongoing work:
  - Couple OpenMP B2 with MPI EIRENE
  - Real world test cases
  - Further improvements to reach bottleneck

# Unit test generator



```
subroutine foo(a, b, c)
    use mydata, only :  matrix
    integer :: a
    real :: b
    real, dimension(:) :: c
    ....
end subroutine foo
```

1. Forcheck

| VARIABLE | TYPE | RANK | SIZE | OPERATIONS | LINES |
|---|---|---|---|---|---|
| A | I | | 4 | P O | 25 27 |
| B | R | | 4 | P O | 25 28 |
| C | R | 1 | | P O | 25 29 |
| MATRIX | R | 2 | | O | 26 32 |

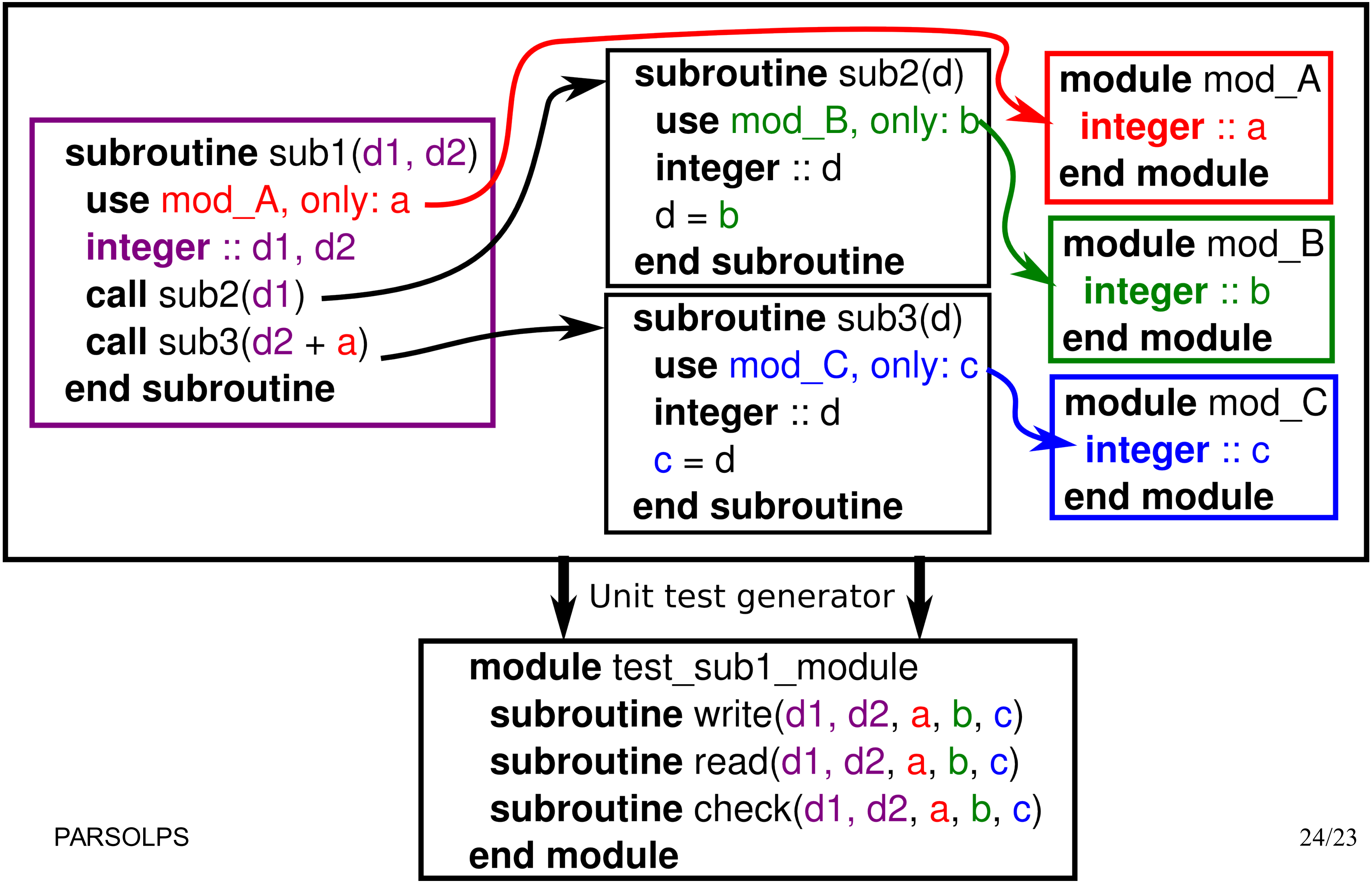
2. python unit test generator

```
#ifdef UNITTEST_SAVE_DATA
        call write_foo(A, B, C, lbound(C), ubound(C),
    &  MATRIX, lbound(MATRIX), ubound(MATRIX))
#endif
```

```
program test_foo
  use test_foo_module
  ...
  call setvar_foo(A, B, C, MATRIX)
  call foo(a, b, c)
  ...
  check_results(A, B, C, MATRIX)
  ....
end program
```

```
module test_foo_module
  public :: init, read_foo, write_foo
  public :: setvar_foo
  public :: store_results, check_results
...
end module
```

# Test side effects



```
subroutine sub1(d1, d2)
  use mod_A, only: a
  integer :: d1, d2
  call sub2(d1)
  call sub3(d2 + a)
end subroutine
```

```
subroutine sub2(d)
  use mod_B, only: b
  integer :: d
  d = b
end subroutine
```

```
subroutine sub3(d)
  use mod_C, only: c
  integer :: d
  c = d
end subroutine
```

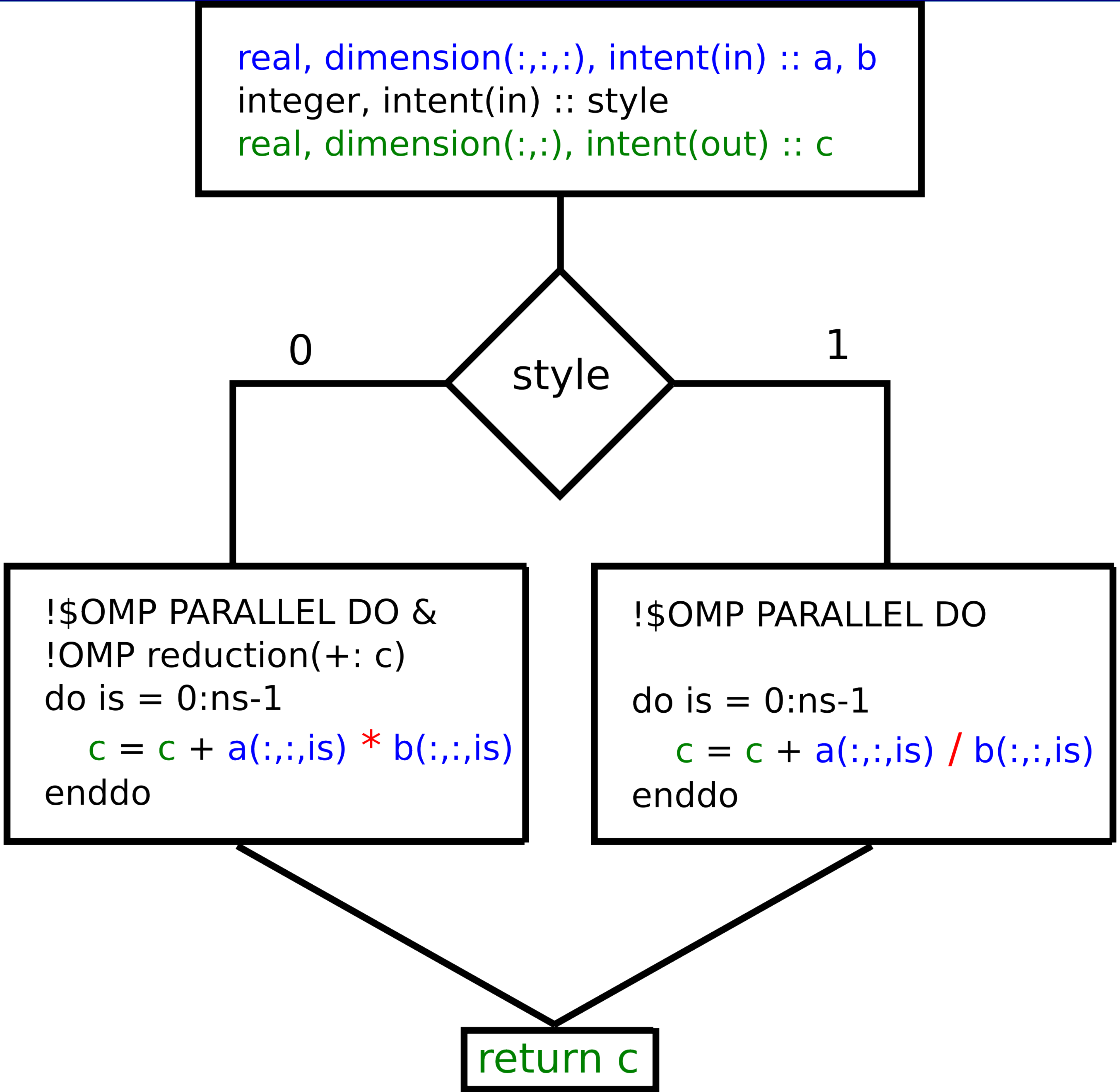
```
module mod_A
  integer :: a
end module
```

```
module mod_B
  integer :: b
end module
```

```
module mod_C
  integer :: c
end module
```

```
module test_sub1_module
  subroutine write(d1, d2, a, b, c)
  subroutine read(d1, d2, a, b, c)
  subroutine check(d1, d2, a, b, c)
end module
```

# Full coverage



```
real, dimension(:,:,:), intent(in) :: a, b
integer, intent(in) :: style
real, dimension(:,:), intent(out) :: c
```

```
!$OMP PARALLEL DO &
!OMP reduction(+: c)
do is = 0:ns-1
    c = c + a(:,:,is) * b(:,:,is)
enddo
```

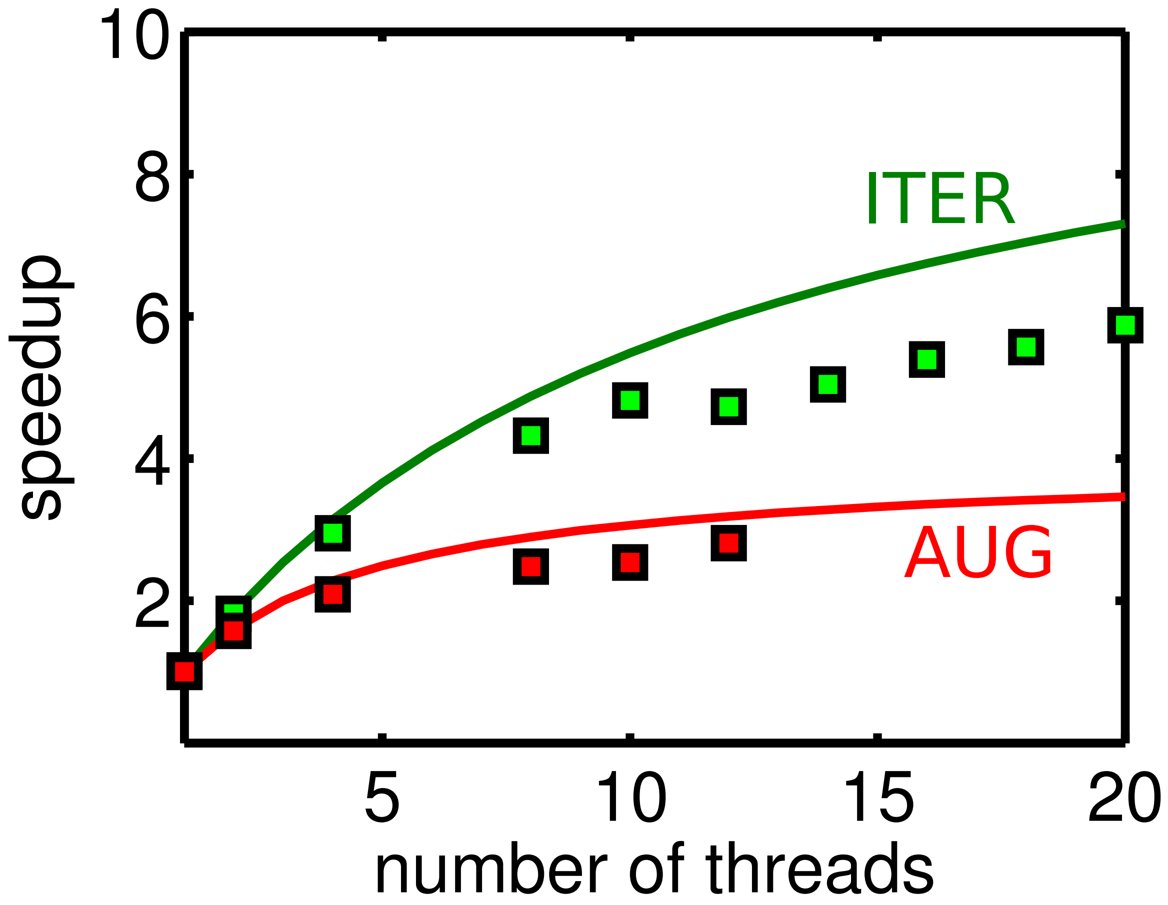
```
!$OMP PARALLEL DO

do is = 0:ns-1
    c = c + a(:,:,is) / b(:,:,is)
enddo
```

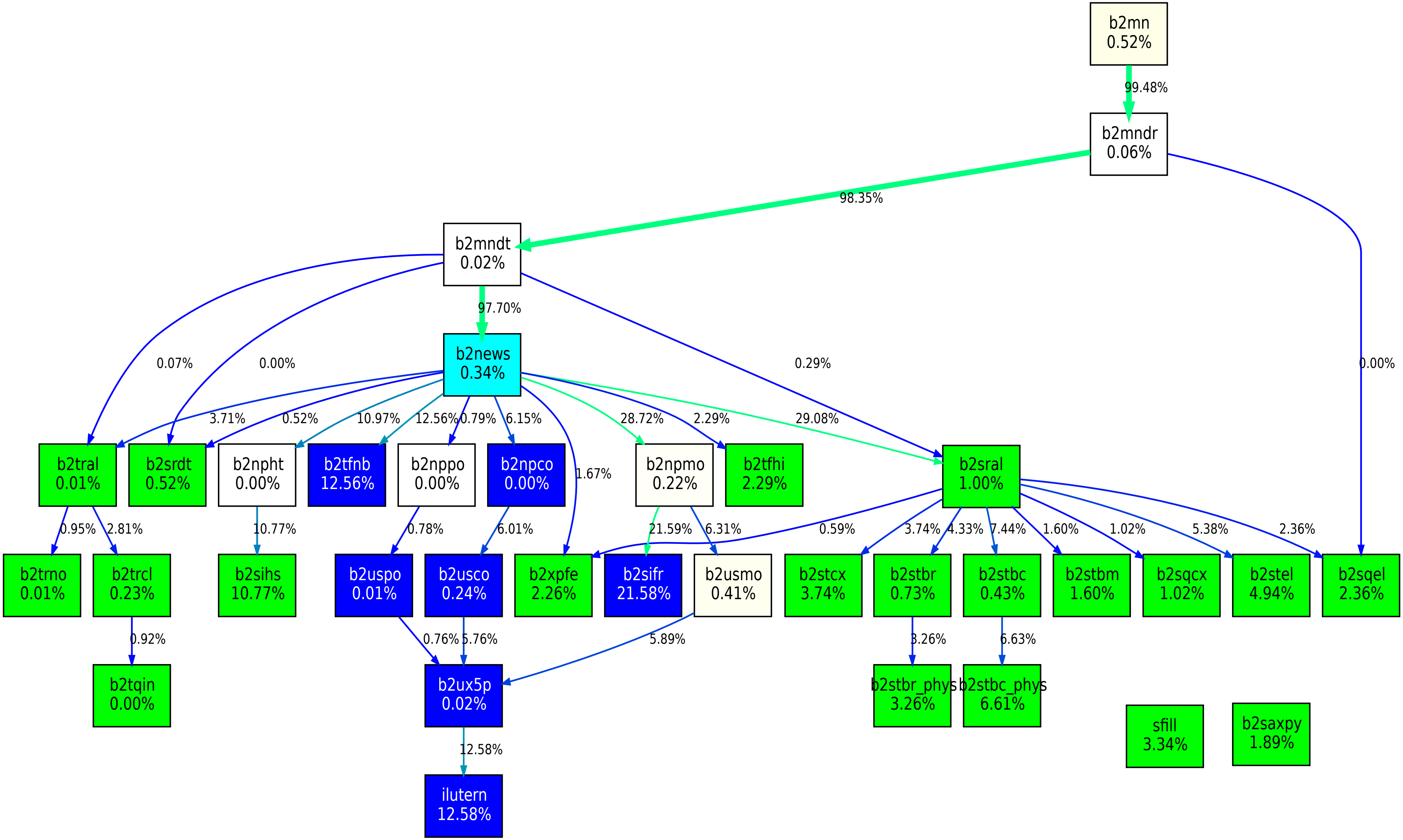
```
return c
```

# B2 Testcases

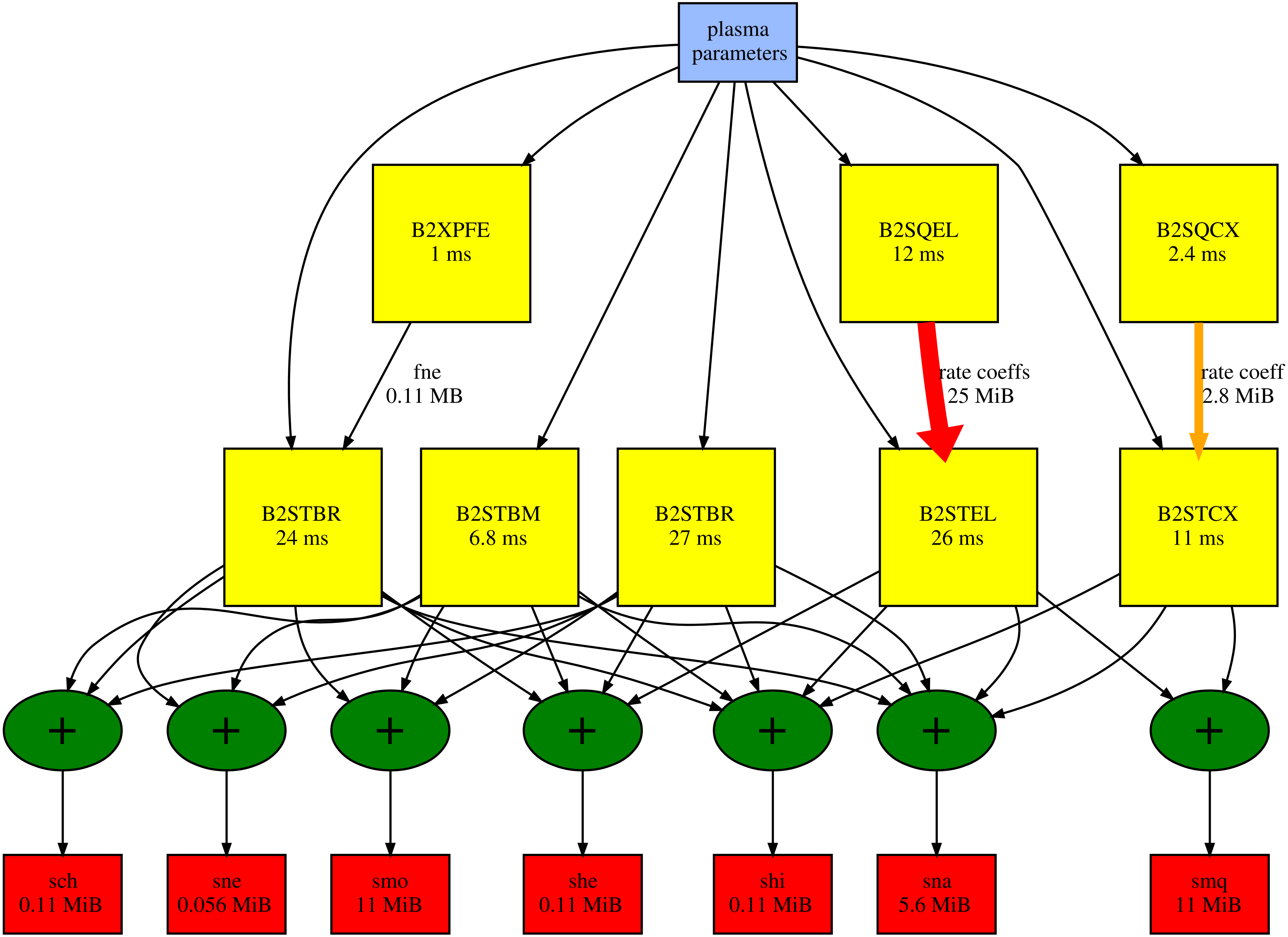
- Parallel fraction depends on simulation parameters
- Timesteps, switches, compiler options
- ITER D+T+He+Be+Ne+W,  AUG_16151_D+C+He



# B2 Callgraph



# Next year: Hybrid MPI-OpenMP


plasma parameters
B2XPFE 1 ms
B2SQEL 12 ms
B2SQCX 2.4 ms
fne 0.11 MB
rate coeffs 25 MiB
rate coeff 2.8 MiB
B2STBR 24 ms
B2STBM 6.8 ms
B2STBR 27 ms
B2STEL 26 ms
B2STCX 11 ms
+
+
+
+
+
+
+
sch 0.11 MiB
sne 0.056 MiB
smo 11 MiB
she 0.11 MiB
shi 0.11 MiB
sna 5.6 MiB
smq 11 MiB