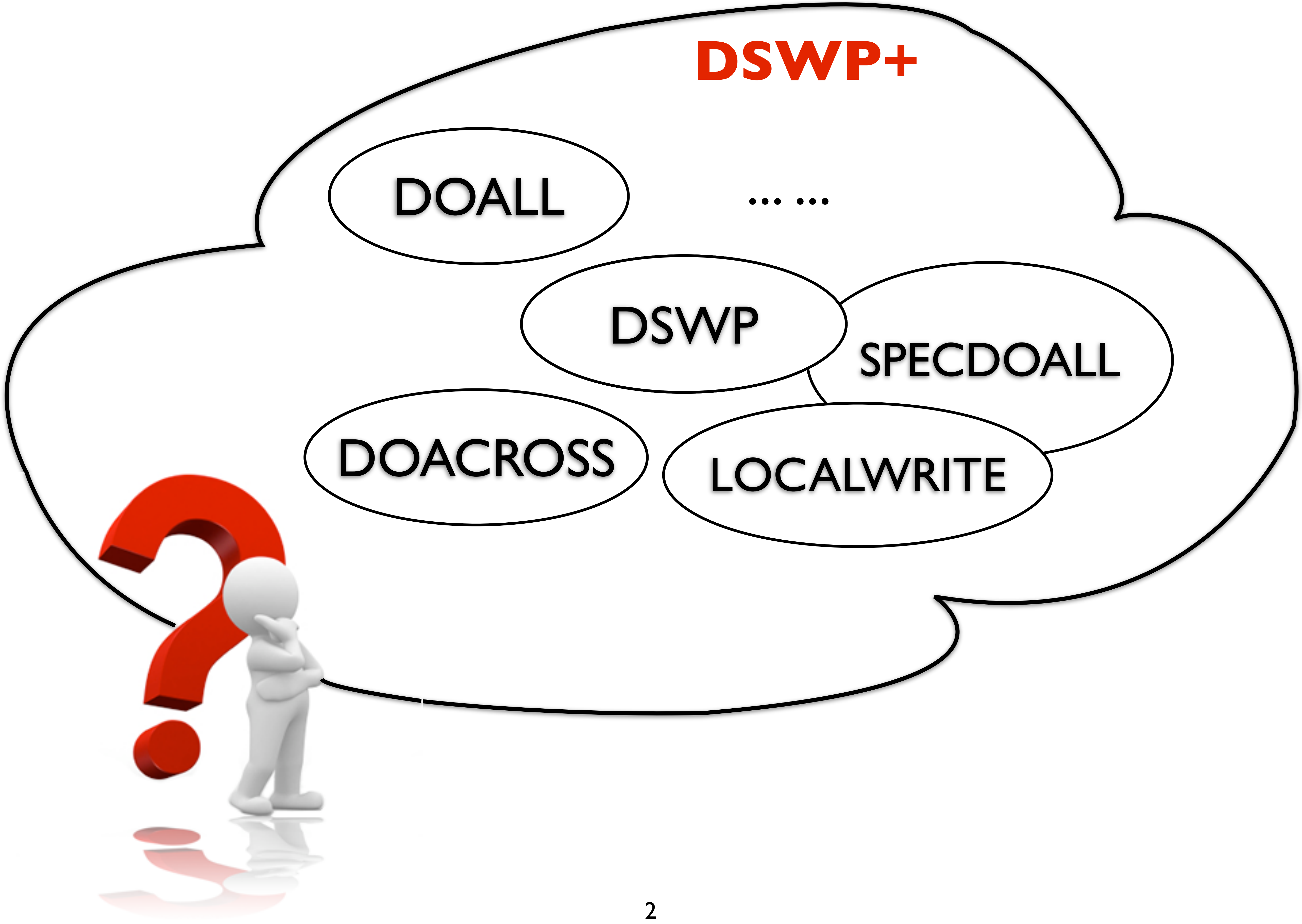
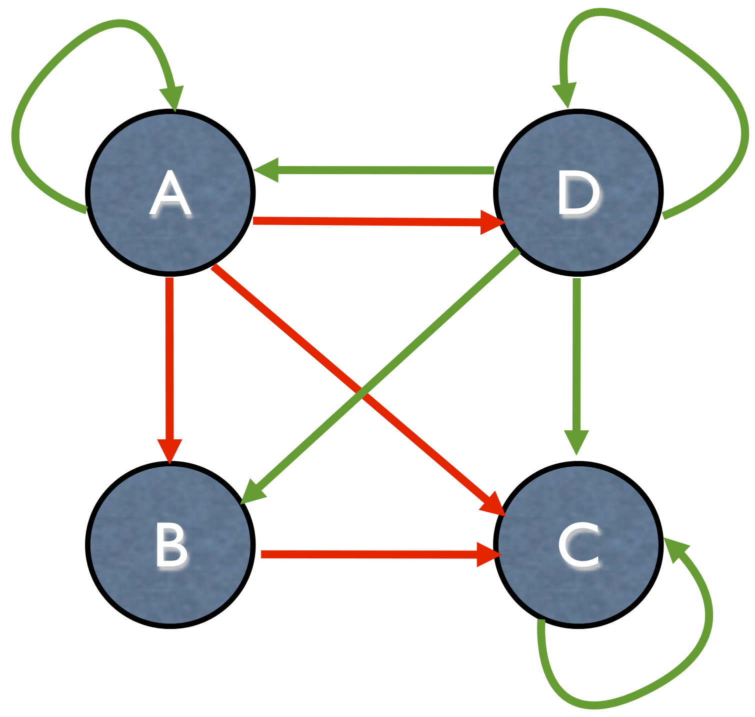
Software Pipelining Creates Parallelization Opportunities

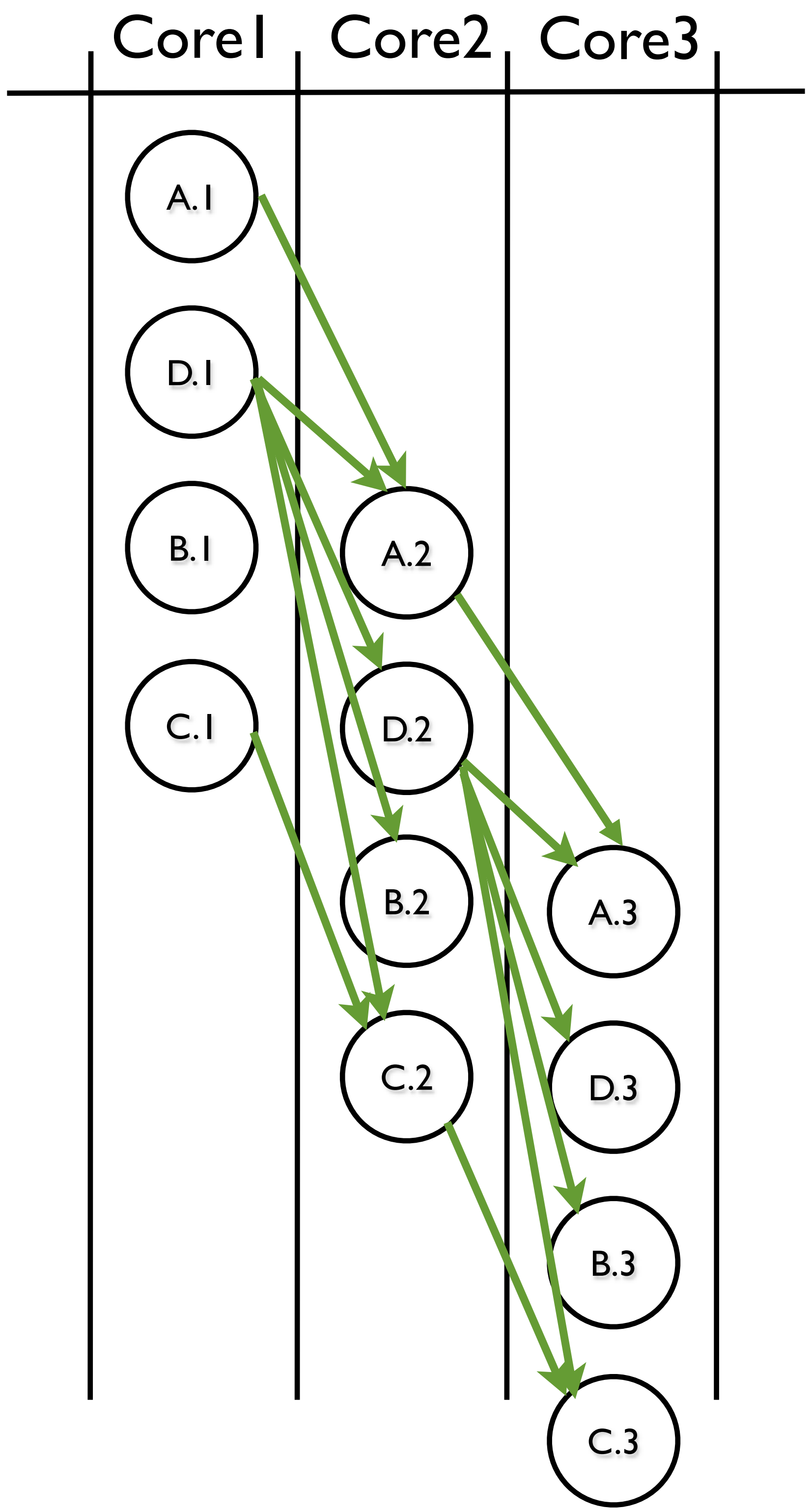
Jialu Huang, Arun Raman, Thomas B. Jablin, Yun Zhang, Tzu-Han Hung
David I. August

Liberty Research Group
Princeton University


DSWP+
DOALL
... ...
DSWP
SPECDOALL
DOACROSS
LOCALWRITE

```
   node = list -> head;
A: while (node != NULL) {
B:     index = calc (node -> data);
C:     density [index] = update_density
           (density [index], node -> data);
D:     node = node -> next;
   }
```



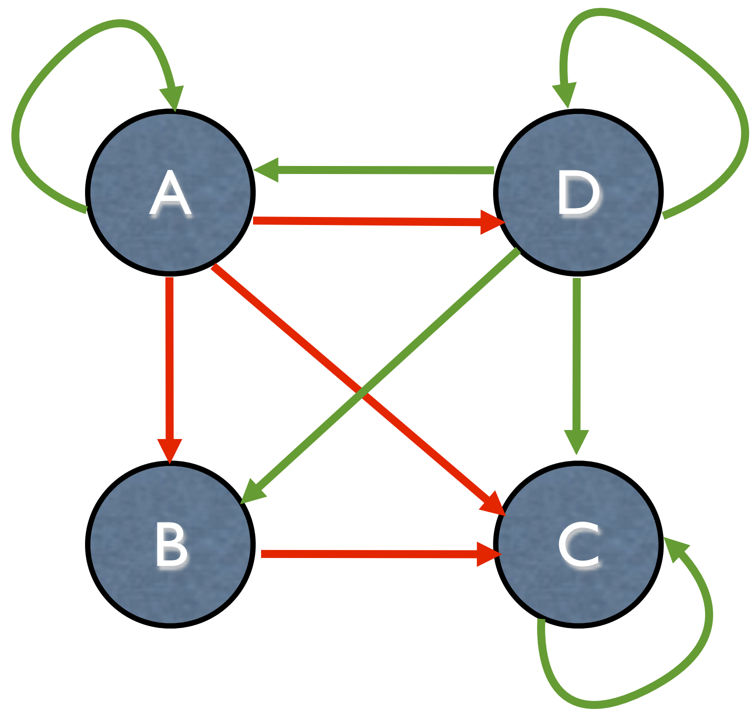


🟥 intra-iteration dependence

🟩 cross-iteration dependence

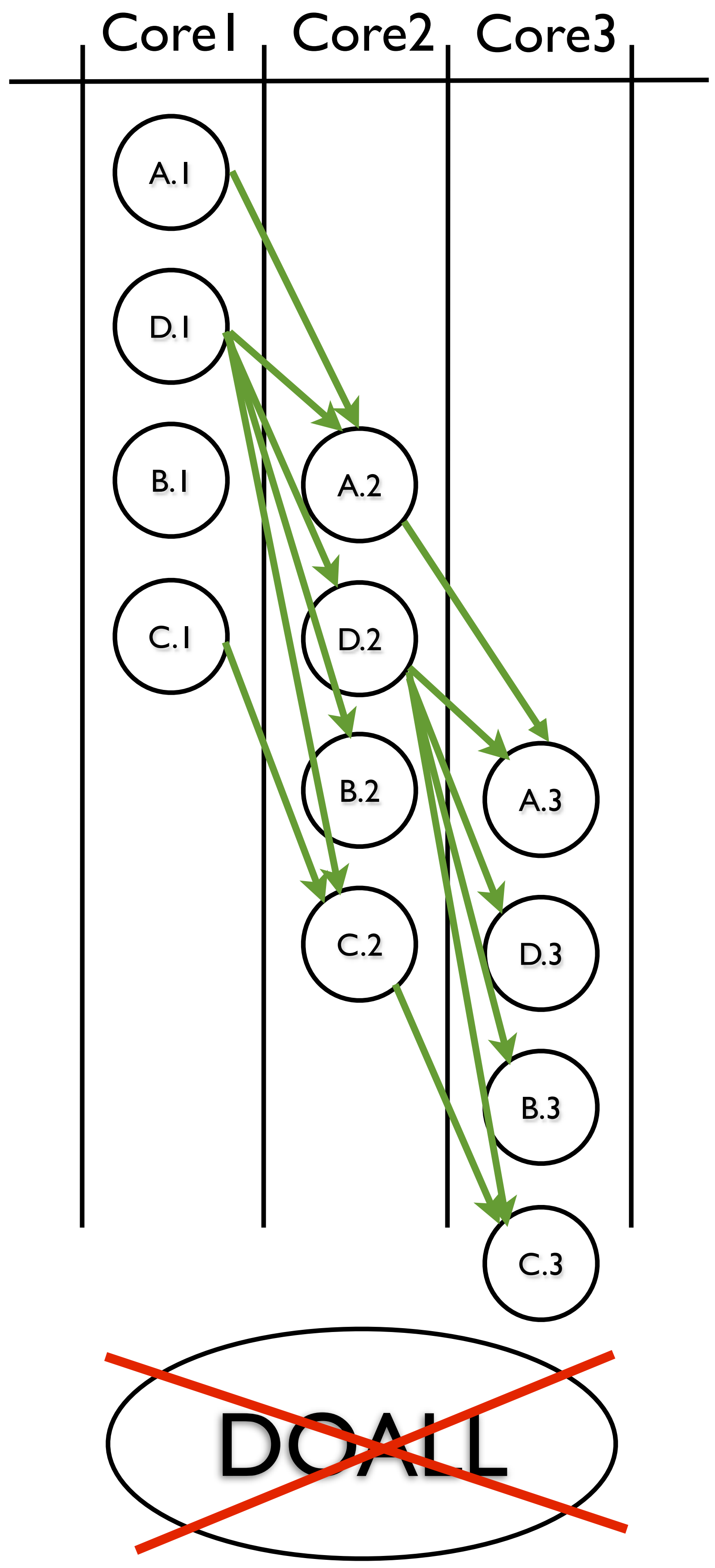
DOALL

```
     node = list -> head;
A:   while (node != NULL) {
B:       index = calc (node -> data);
C:       density [index] = update_density
             (density [index], node -> data);
D:       node = node -> next;
     }
```

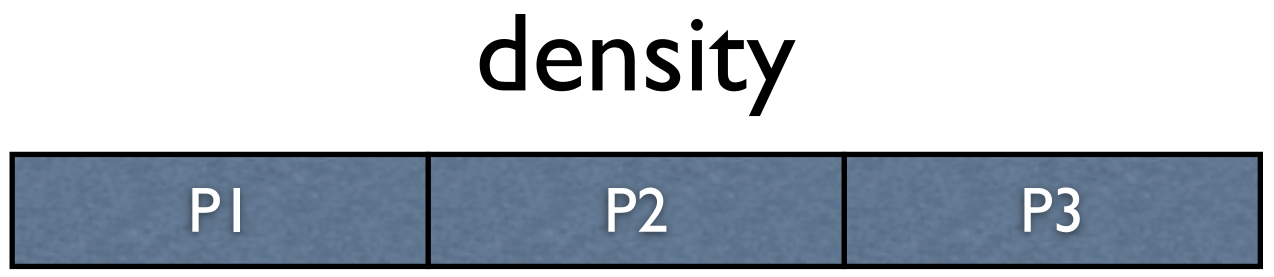


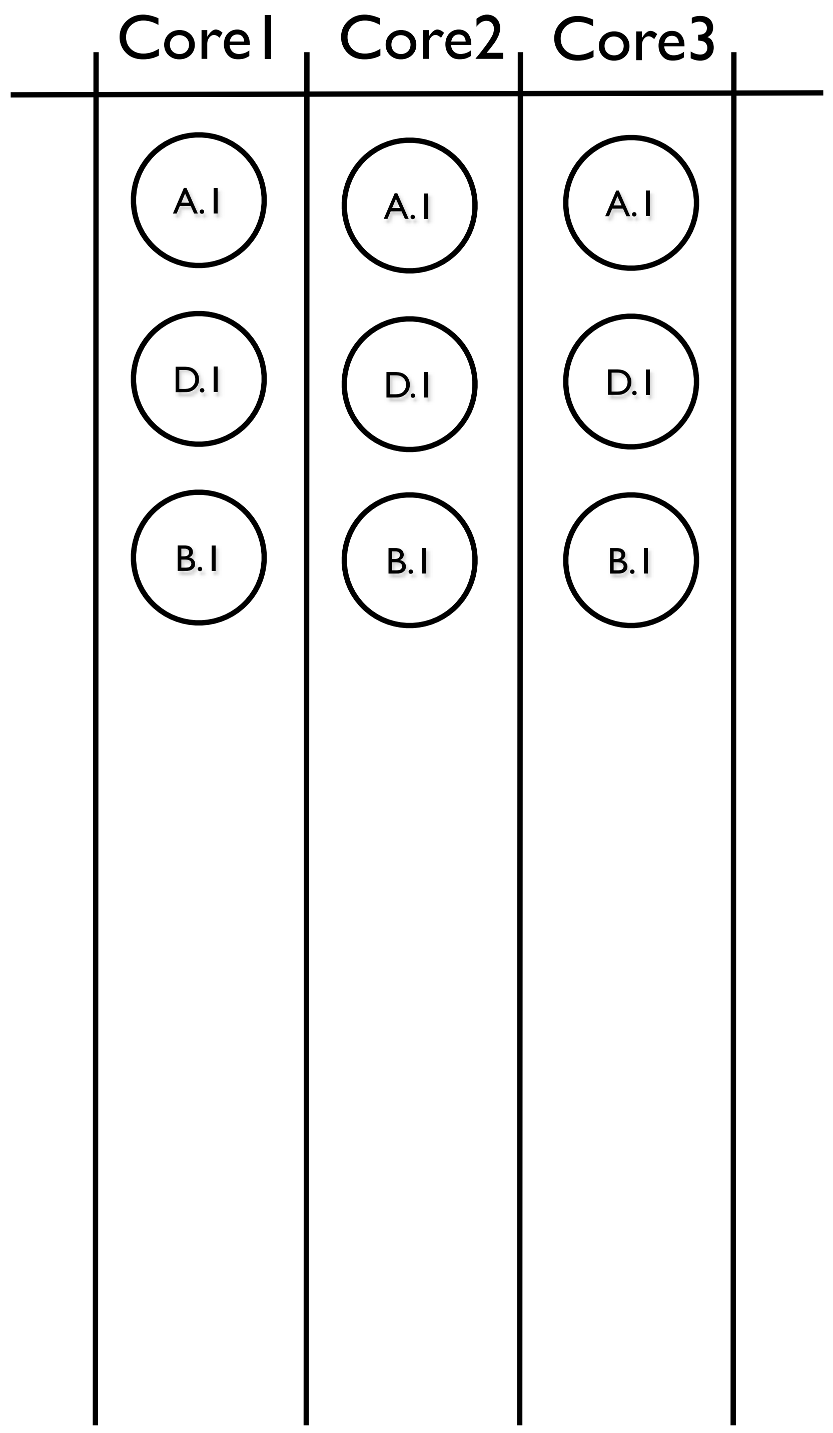
🟥 intra-iteration dependence

🟩 cross-iteration dependence

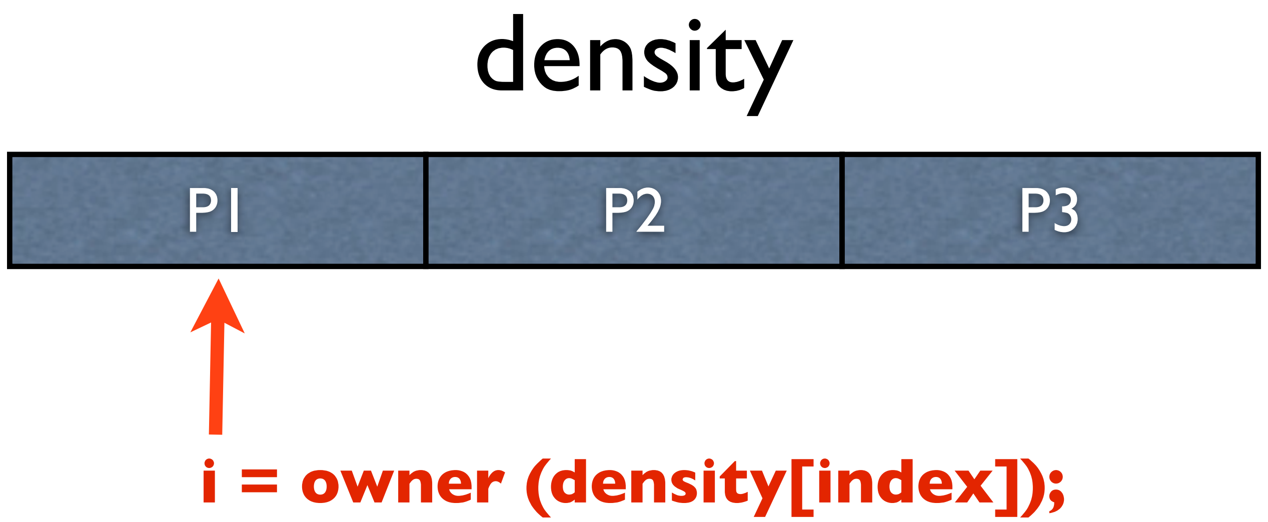


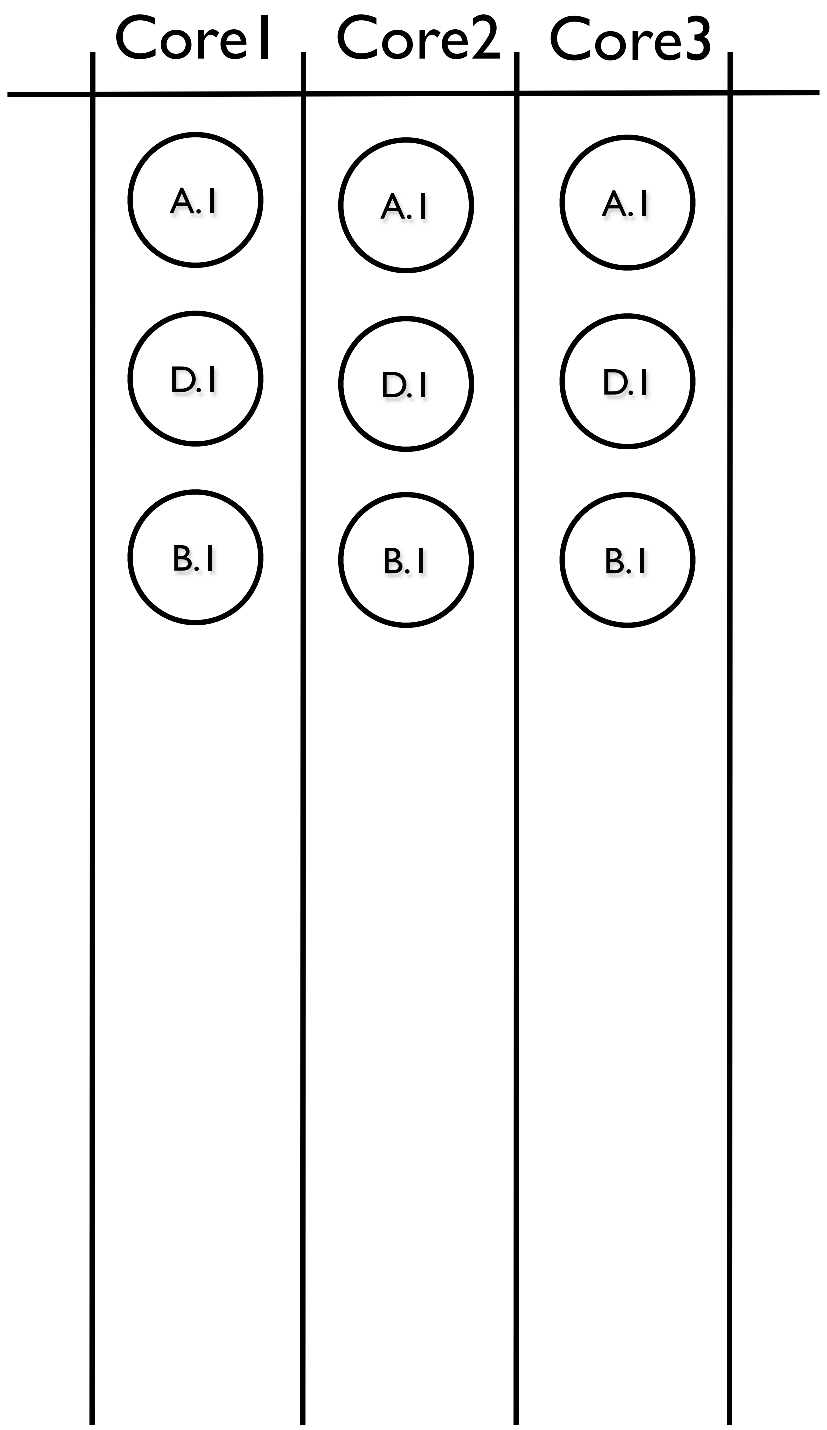
```
    node = list -> head;
A:  while (node != NULL) {
B:      index = calc (node -> data);
C:      density [index] = update_density
            (density [index], node -> data);
D:      node = node -> next;
    }
```





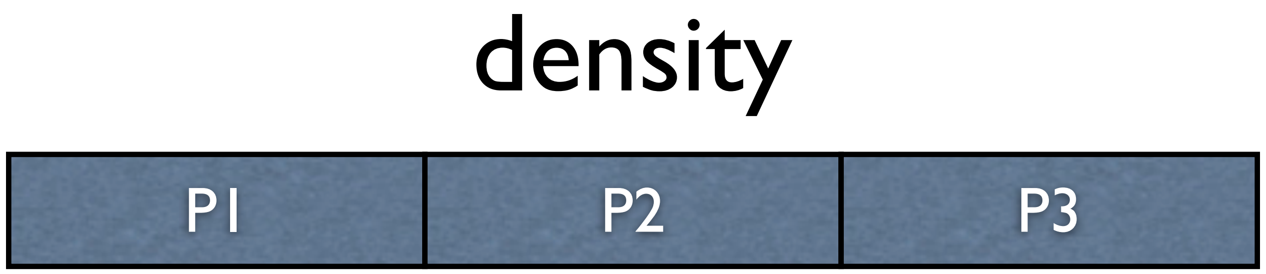
```
    node = list -> head;
A:  while (node != NULL) {
B:      index = calc (node -> data);
C:      density [index] = update_density
            (density [index], node -> data);
D:      node = node -> next;
    }
```

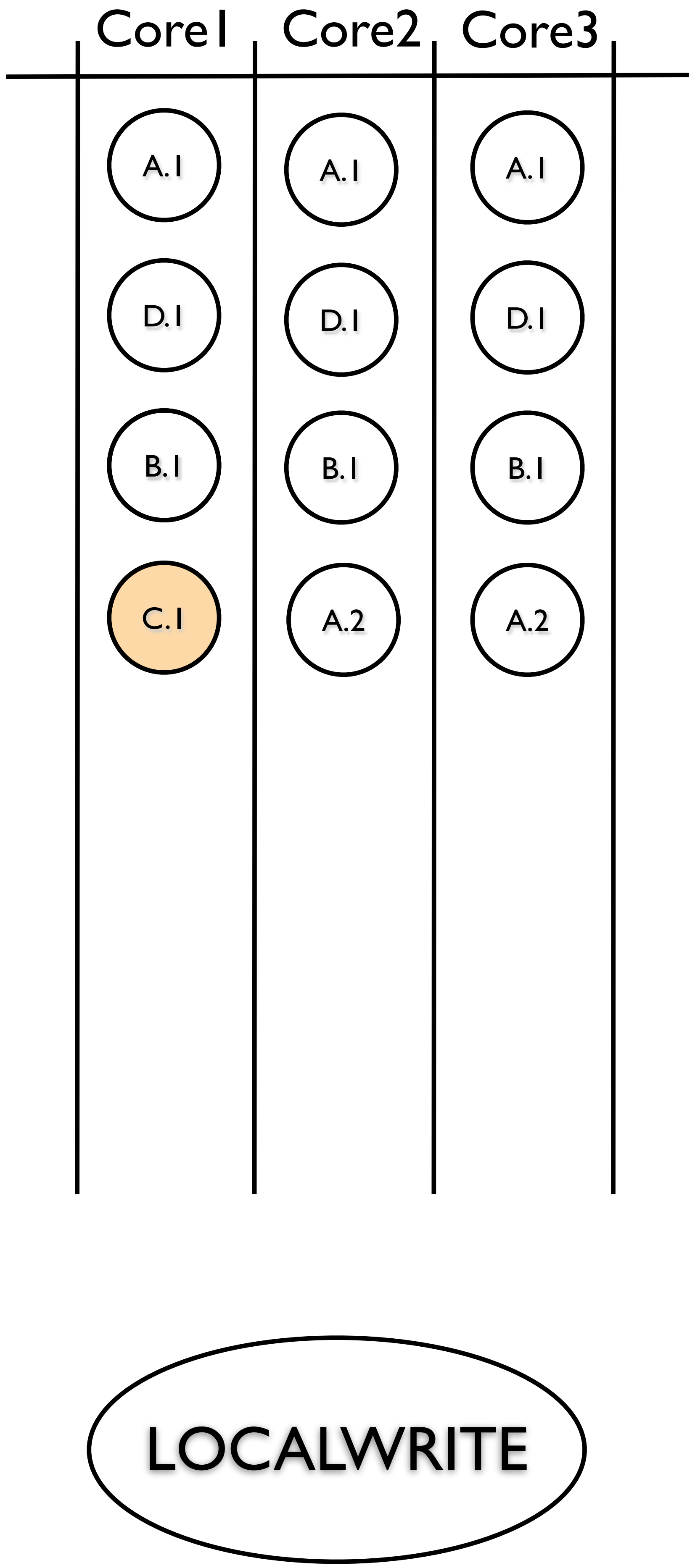




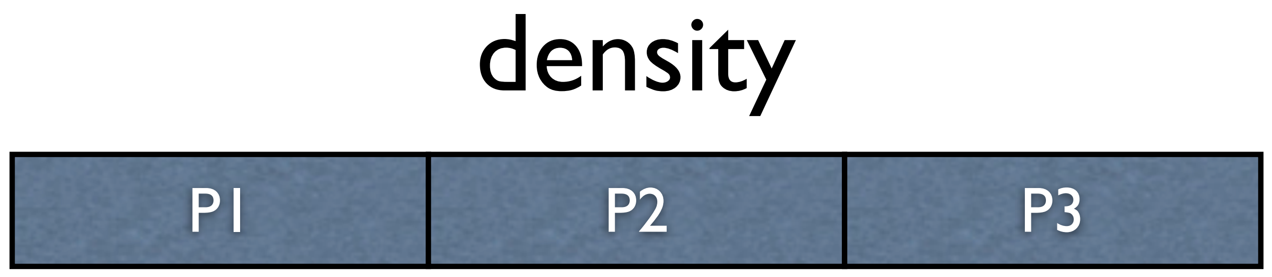
LOCALWRITE

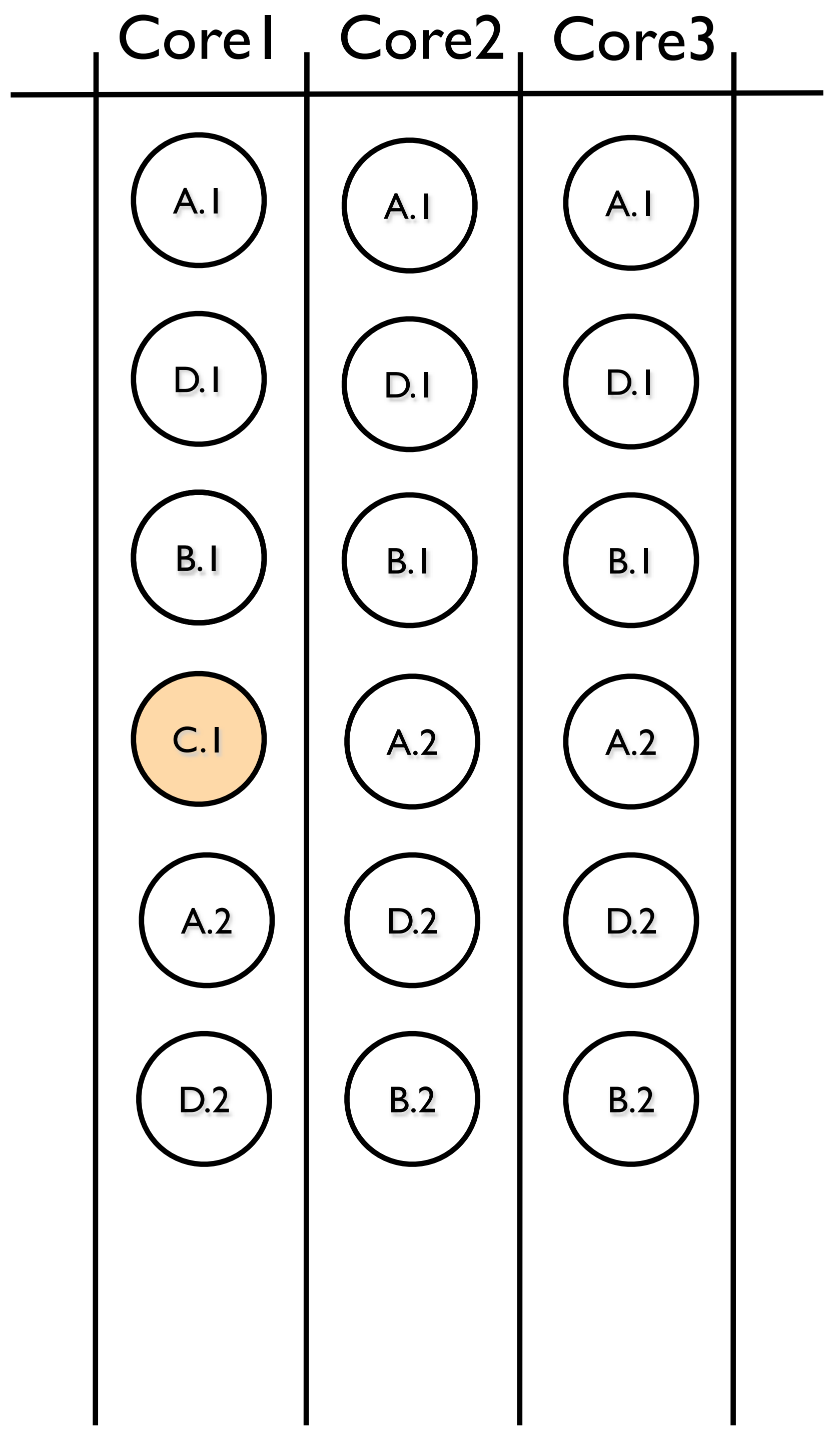
```
     node = list -> head;
A:   while (node != NULL) {
B:        index = calc (node -> data);
C:        density [index] = update_density
              (density [index], node -> data);
D:        node = node -> next;
     }
```





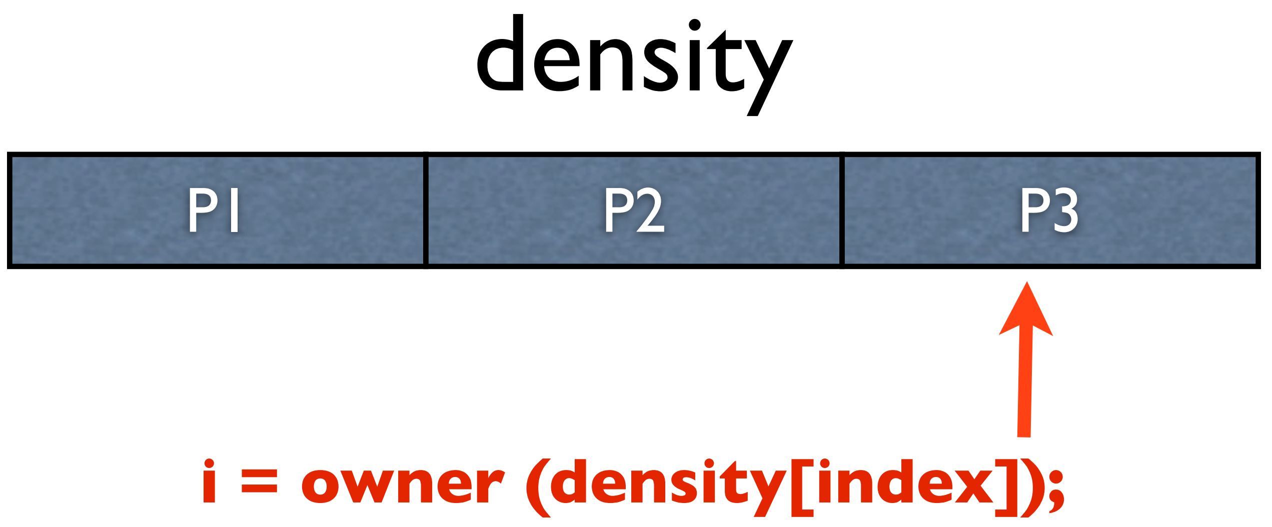
```
    node = list -> head;
A:  while (node != NULL) {
B:      index = calc (node -> data);
C:      density [index] = update_density
            (density [index], node -> data);
D:      node = node -> next;
    }
```

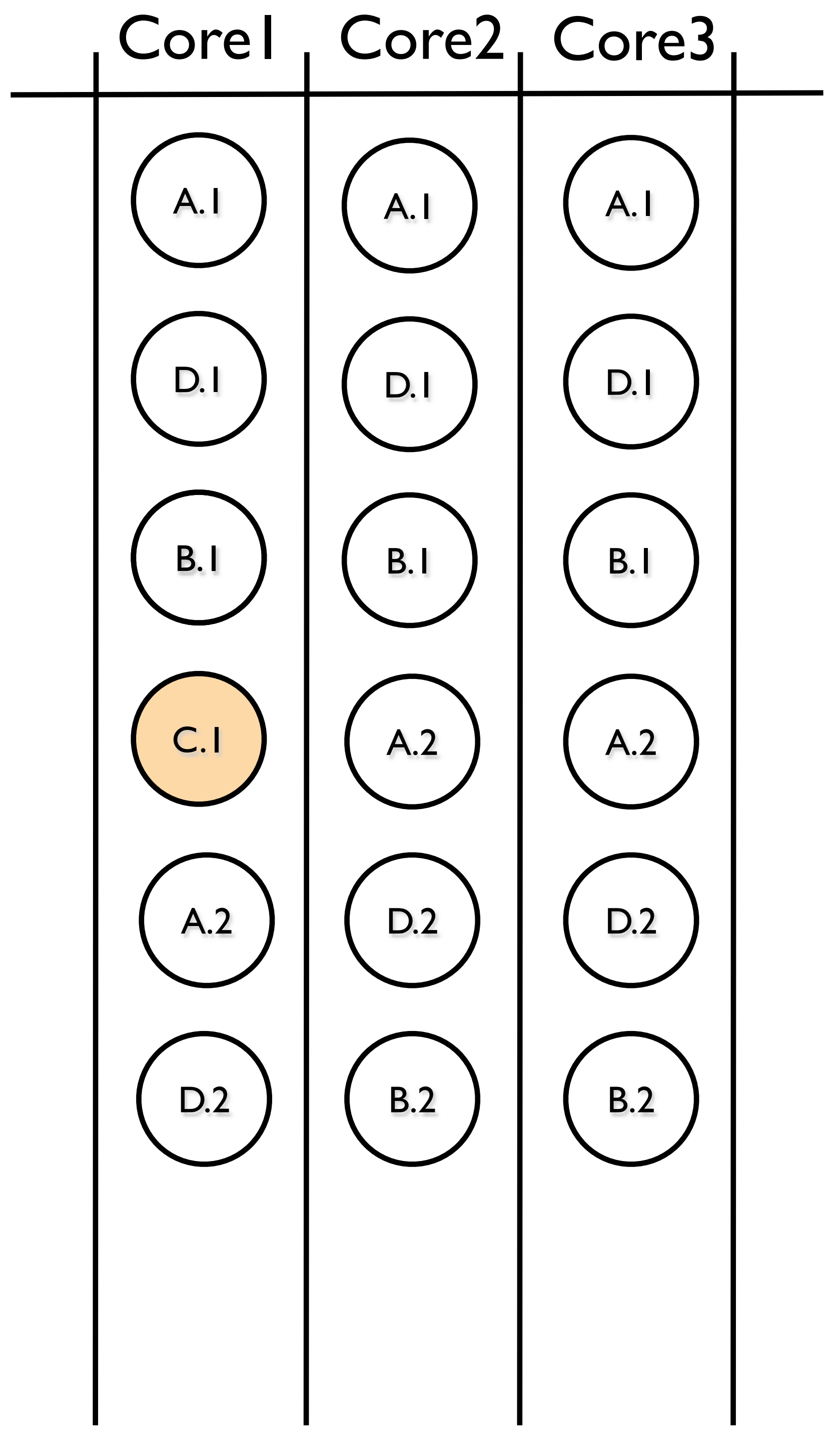




LOCALWRITE

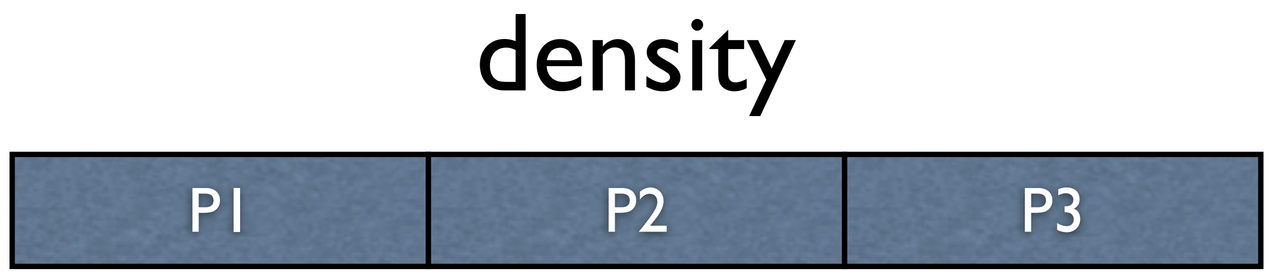
```
    node = list -> head;
A:  while (node != NULL) {
B:      index = calc (node -> data);
C:      density [index] = update_density
            (density [index], node -> data);
D:      node = node -> next;
    }
```

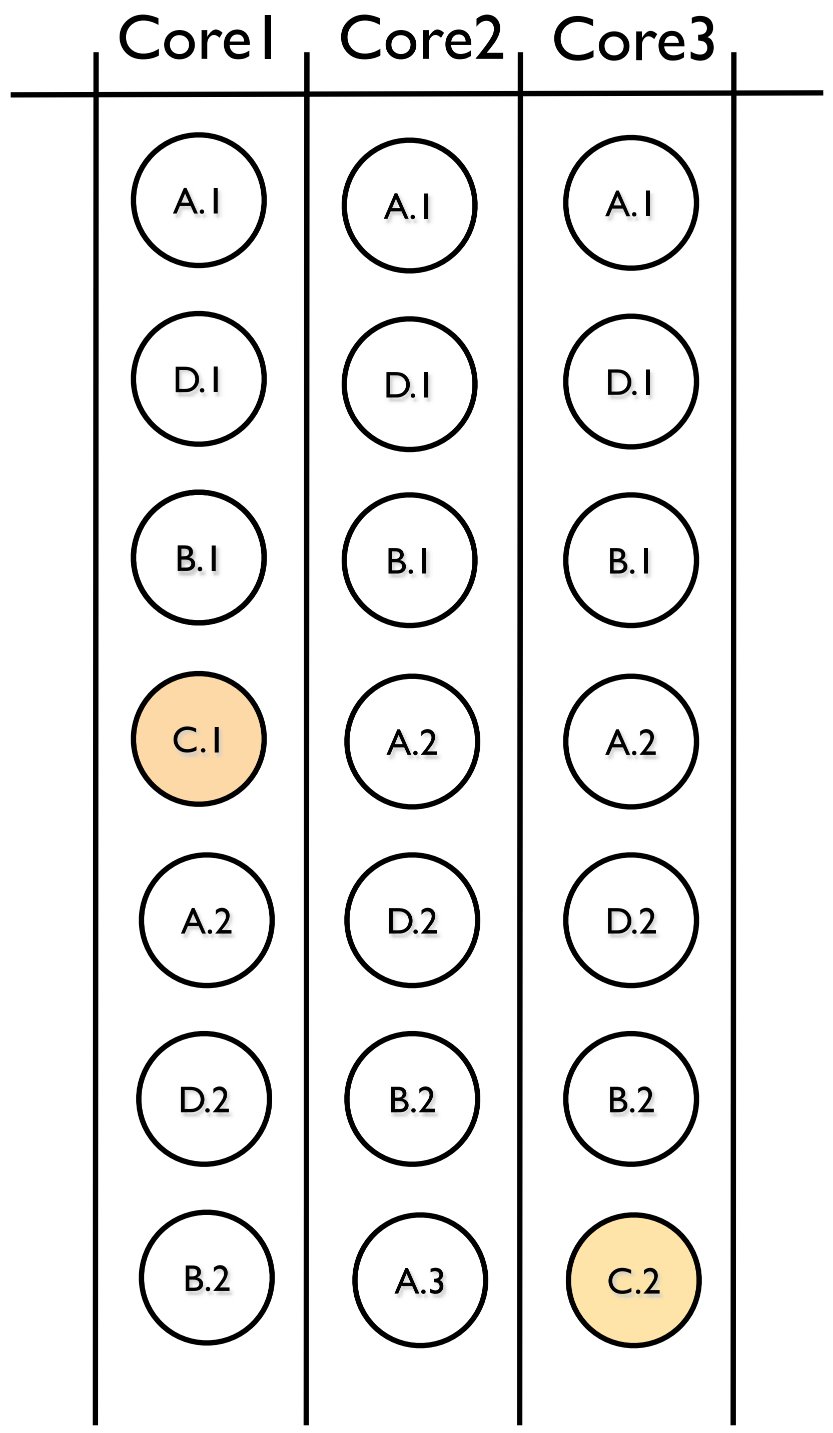




LOCALWRITE

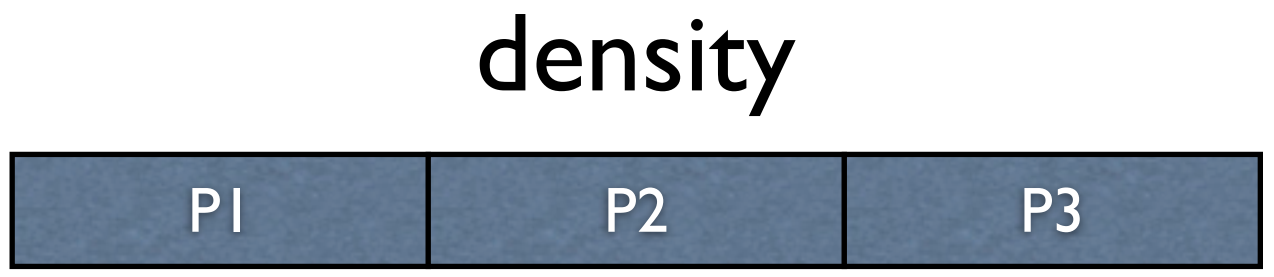
```
    node = list -> head;
A:  while (node != NULL) {
B:      index = calc (node -> data);
C:      density [index] = update_density
            (density [index], node -> data);
D:      node = node -> next;
    }
```

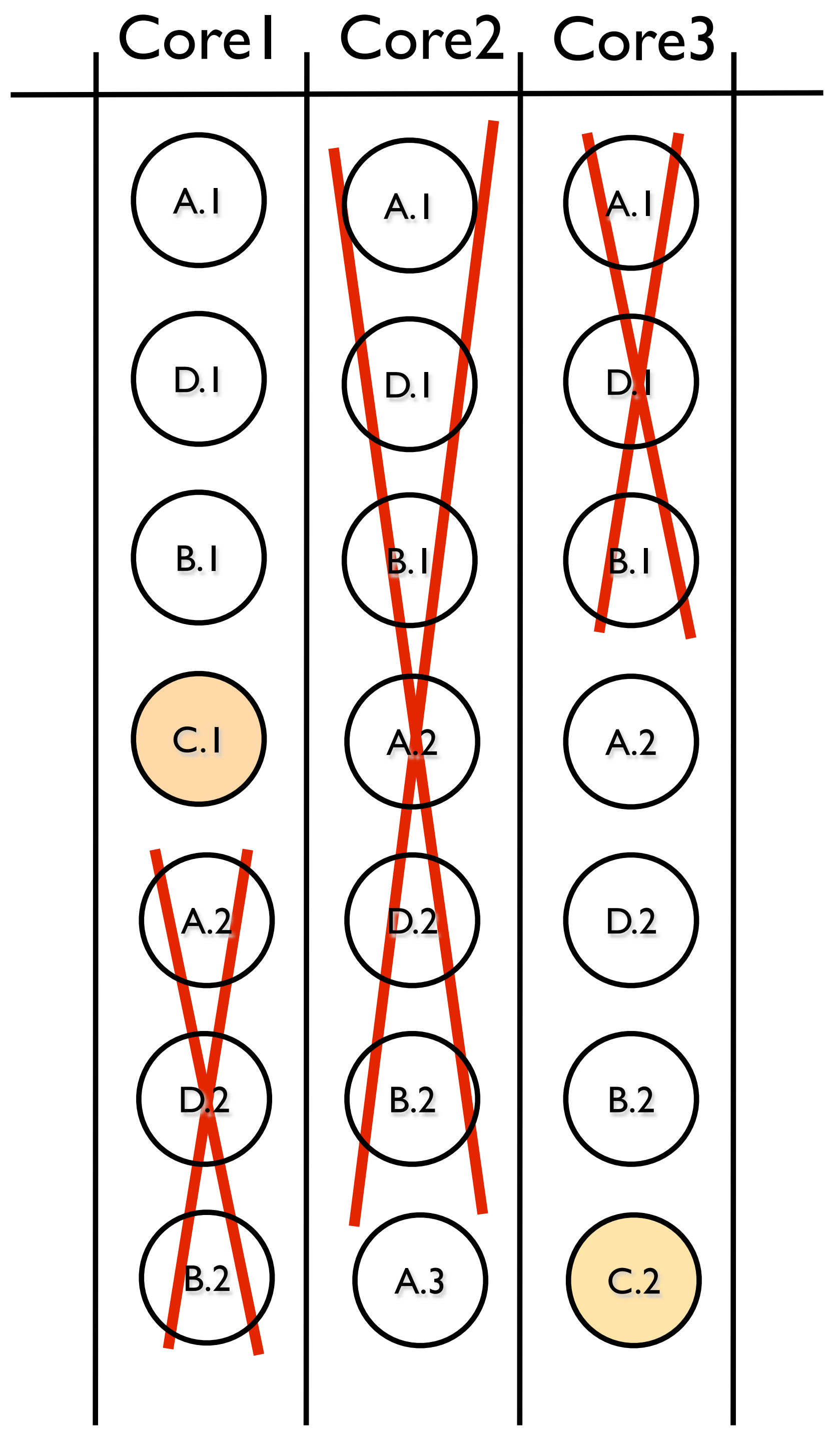




LOCALWRITE

```
     node = list -> head;
A:   while (node != NULL) {
B:       index = calc (node -> data);
C:       density [index] = update_density
             (density [index], node -> data);
D:       node = node -> next;
     }
```

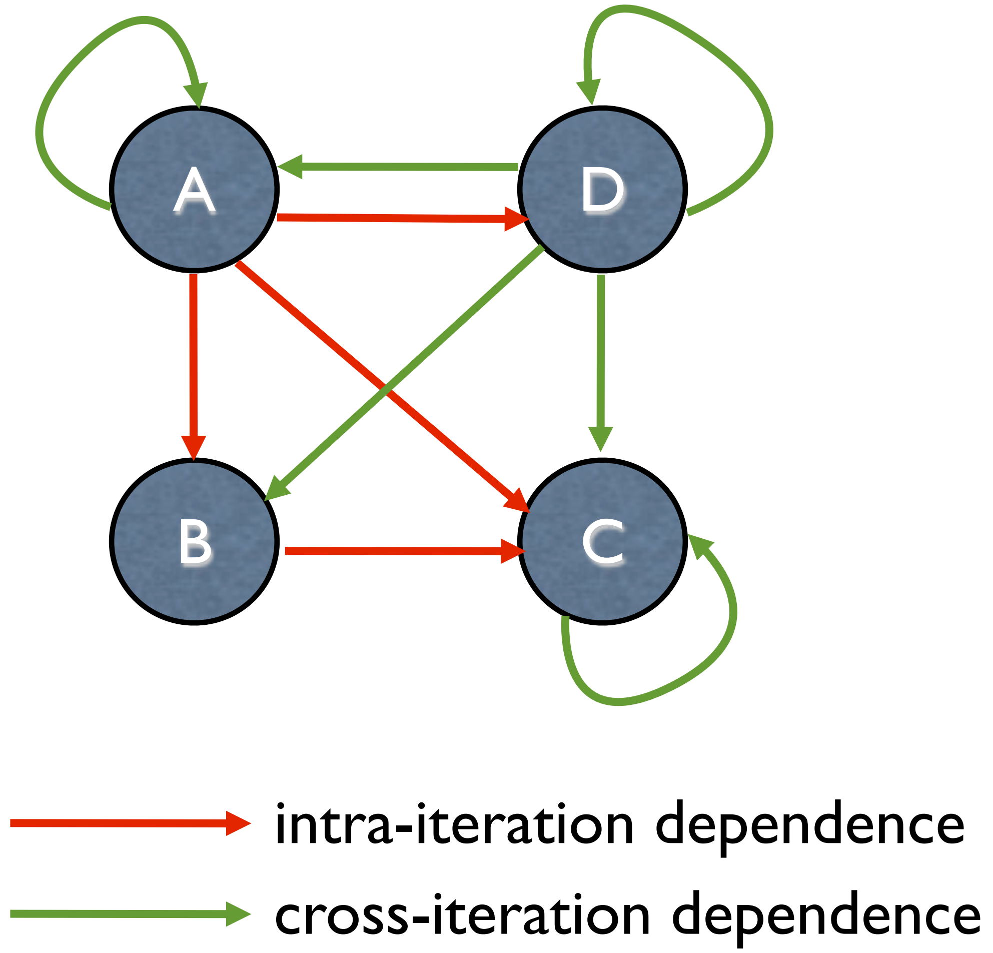




LOCALWRITE

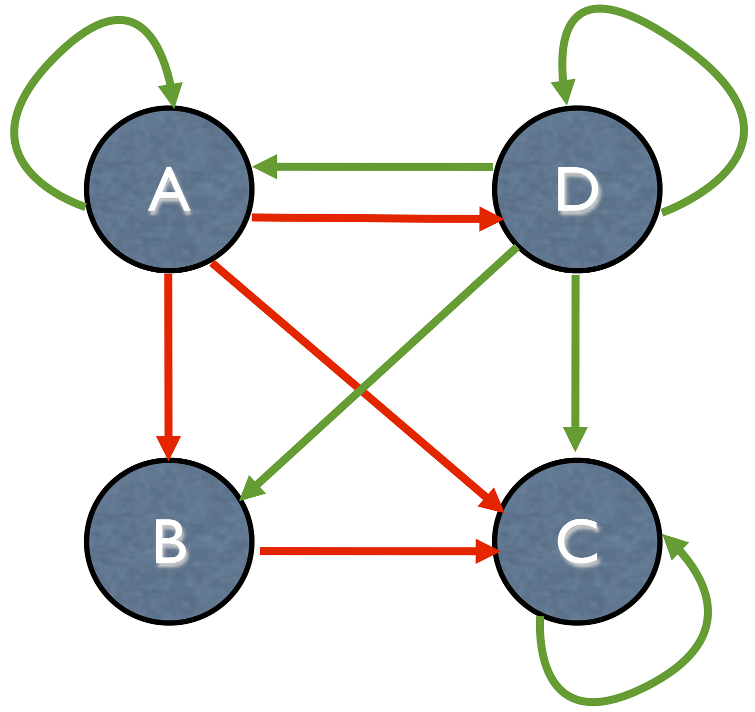
Original Loop:

```
    node = list -> head;
A:  while (node != NULL) {
B:      index = calc (node -> data);
C:      density [index] = update_density
            (density [index], node -> data);
D:      node = node -> next;
    }
```



Original Loop:

```
    node = list -> head;
A:  while (node != NULL) {
B:      index = calc (node -> data);
C:      density [index] = update_density
            (density [index], node -> data);
D:      node = node -> next;
    }
```

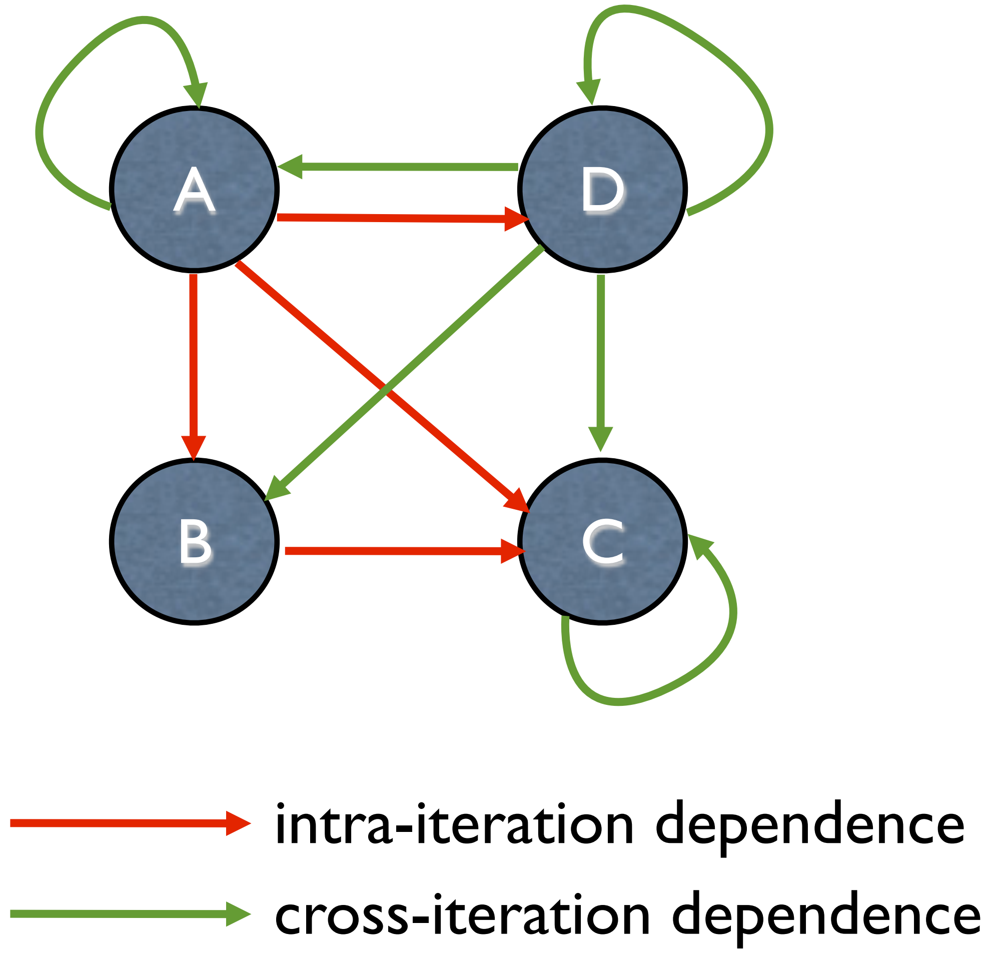


🟥 intra-iteration dependence

🟩 cross-iteration dependence

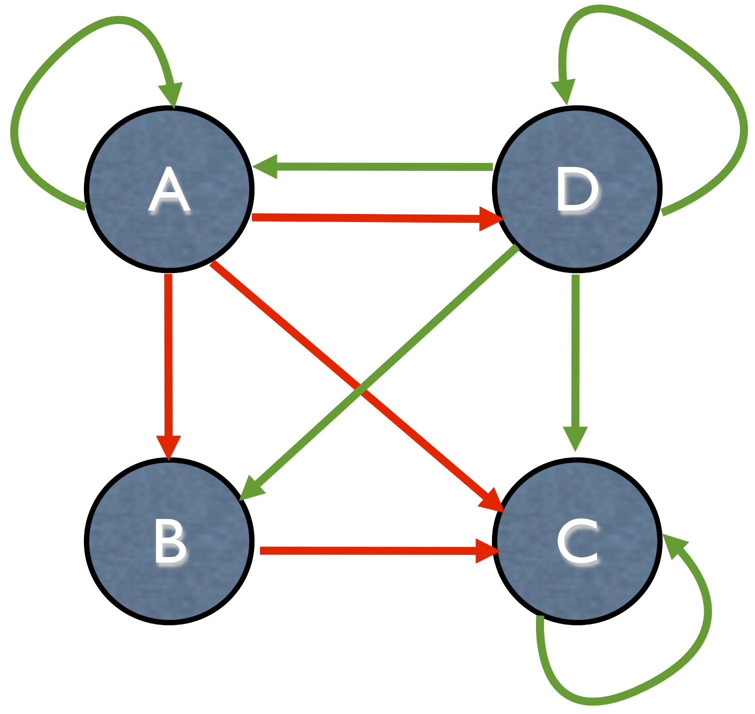
Original Loop:

```
     node = list -> head;
A:   while (node != NULL) {
B:       index = calc (node -> data);
C:       density [index] = update_density
             (density [index], node -> data);
D:       node = node -> next;
     }
```



Original Loop:

```
    node = list -> head;
A:  while (node != NULL) {
B:      index = calc (node -> data);
C:      density [index] = update_density
            (density [index], node -> data);
D:      node = node -> next;
    }
```



🟥 intra-iteration dependence

🟩 cross-iteration dependence

After Partition:

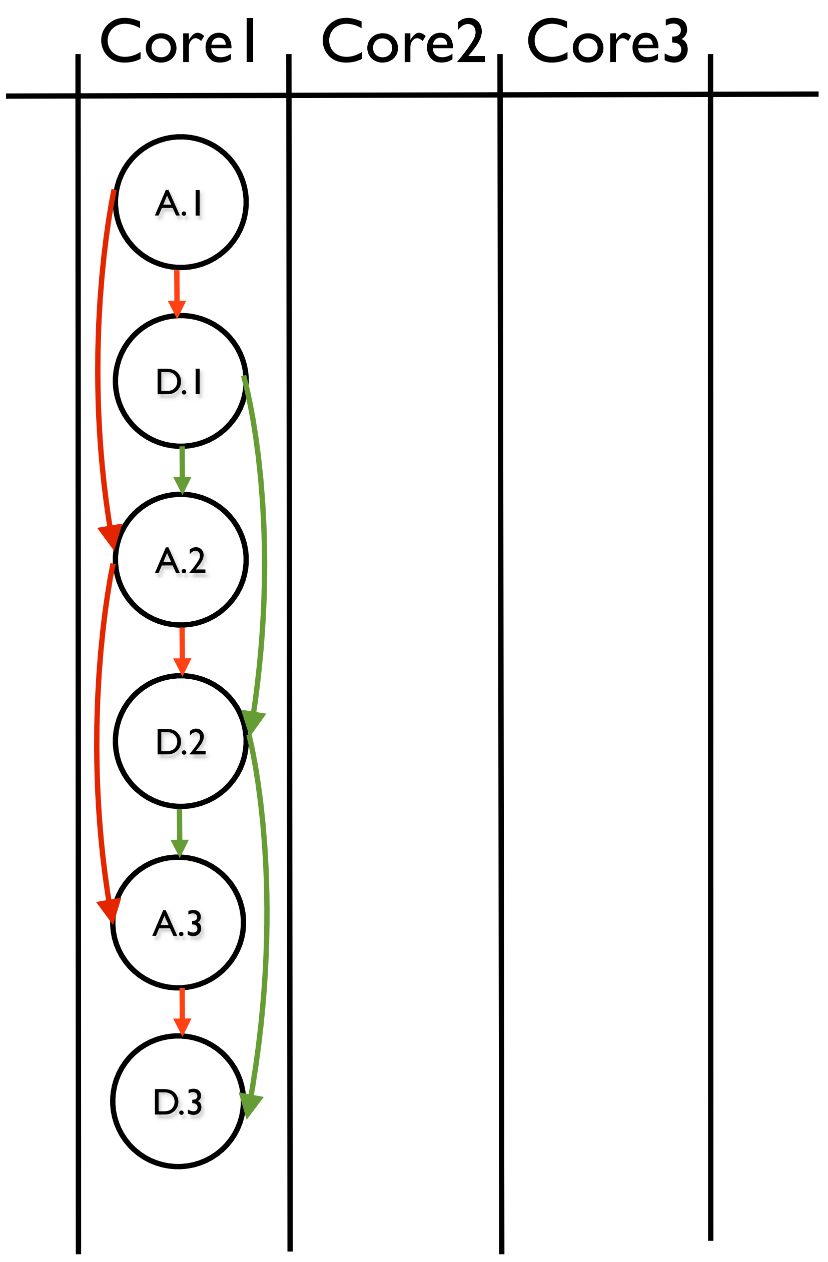
```
    node = list -> head;
A:  while (node != NULL) {
D:      node = node -> next;
}
```

```
    while (TRUE) {
E:      node = getNodeOrExit();
B:      index = calc
          (node -> data);
    }
```

```
    while (TRUE) {
F:      node = getNodeOrExit();
G:      index = getIndex();
C:      density [index] = update_density
           ( density [index], node -> data);
    }
```

Sequential

```
    node = list -> head;
A:  while (node != NULL) {
D:      node = node -> next;
}
```

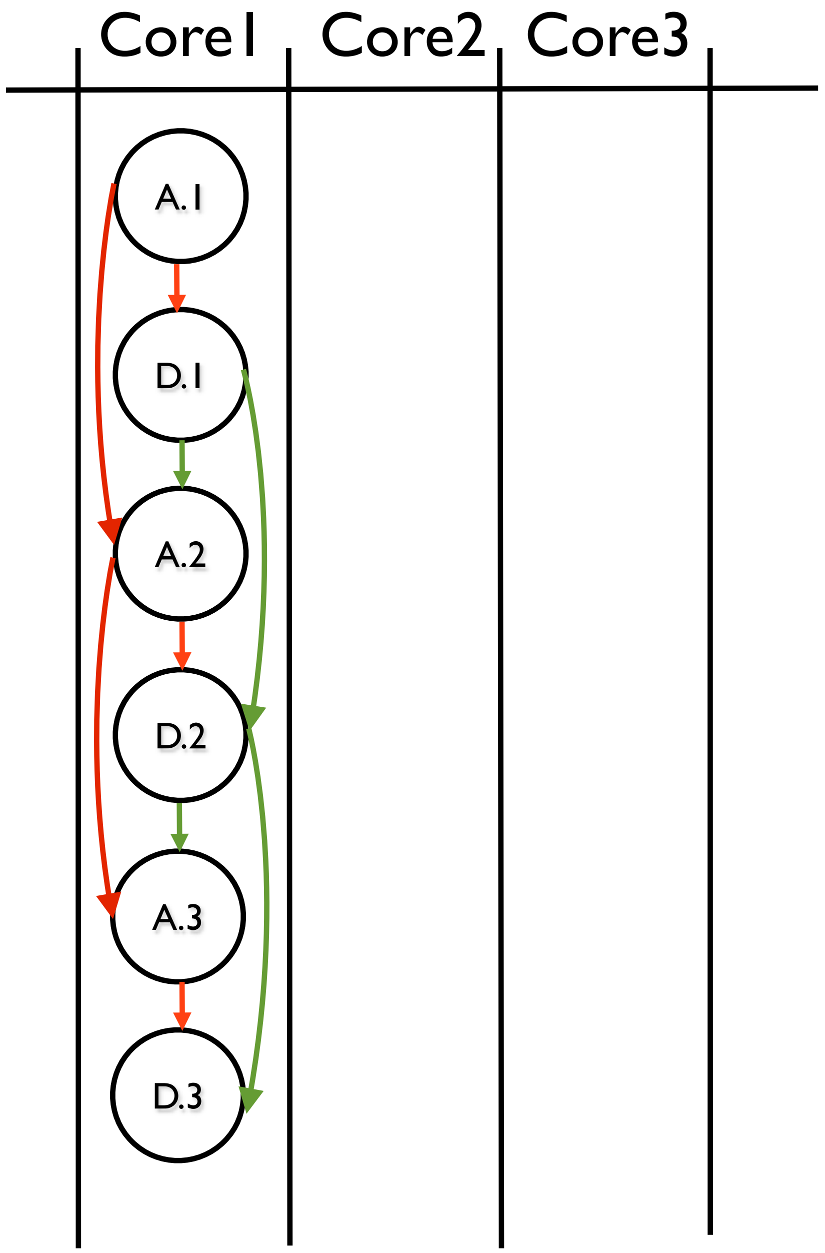


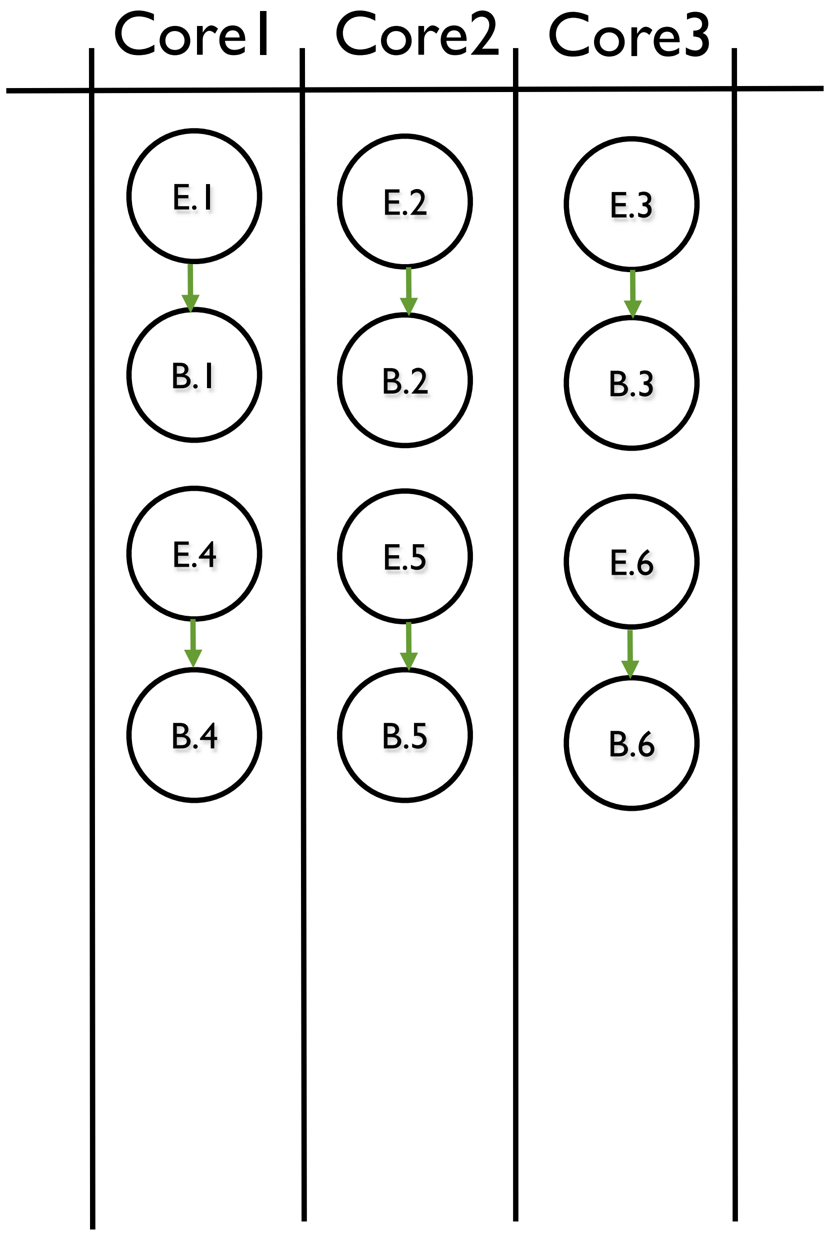
Sequential

```
    node = list -> head;
A:  while (node != NULL) {
D:      node = node -> next;
}
```

DOALL

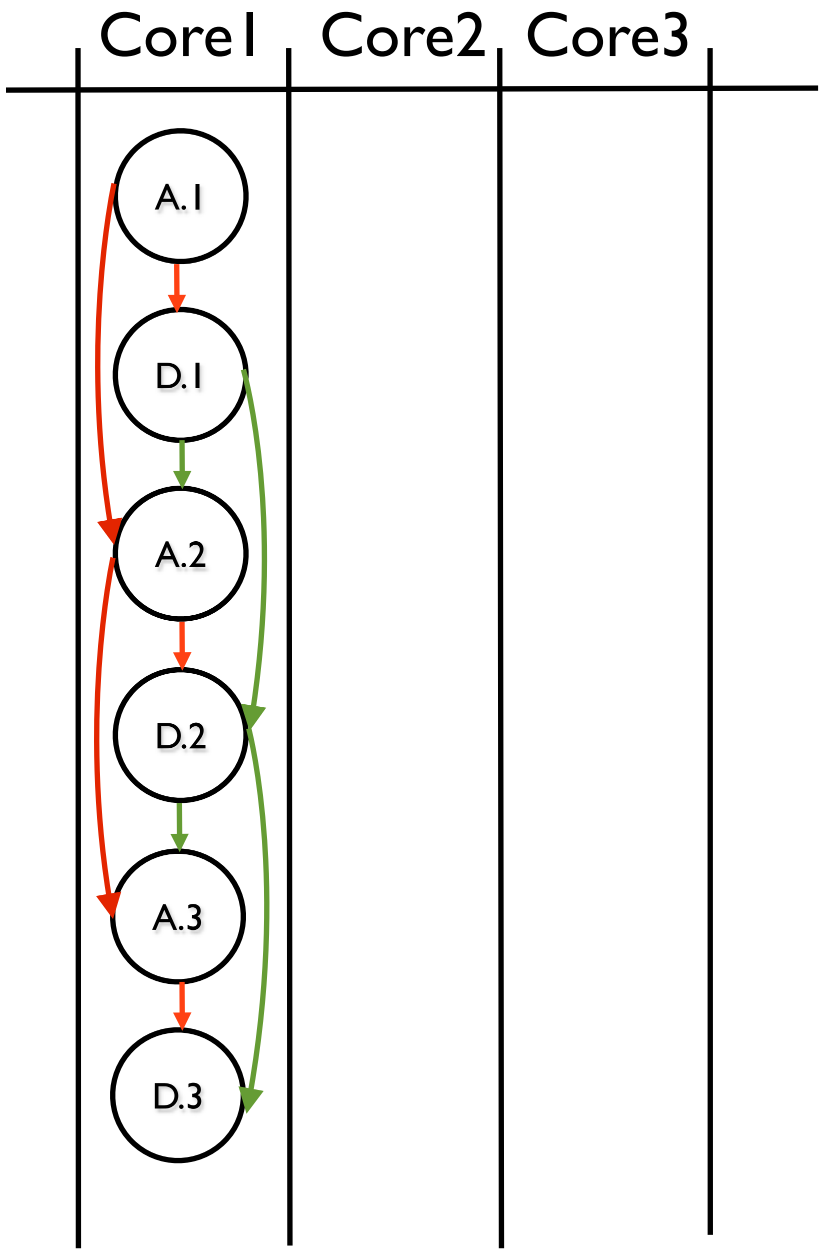
```
    while (TRUE) {
E:      node = getNodeOrExit();
B:      index = calc
           (node -> data);
    }
```





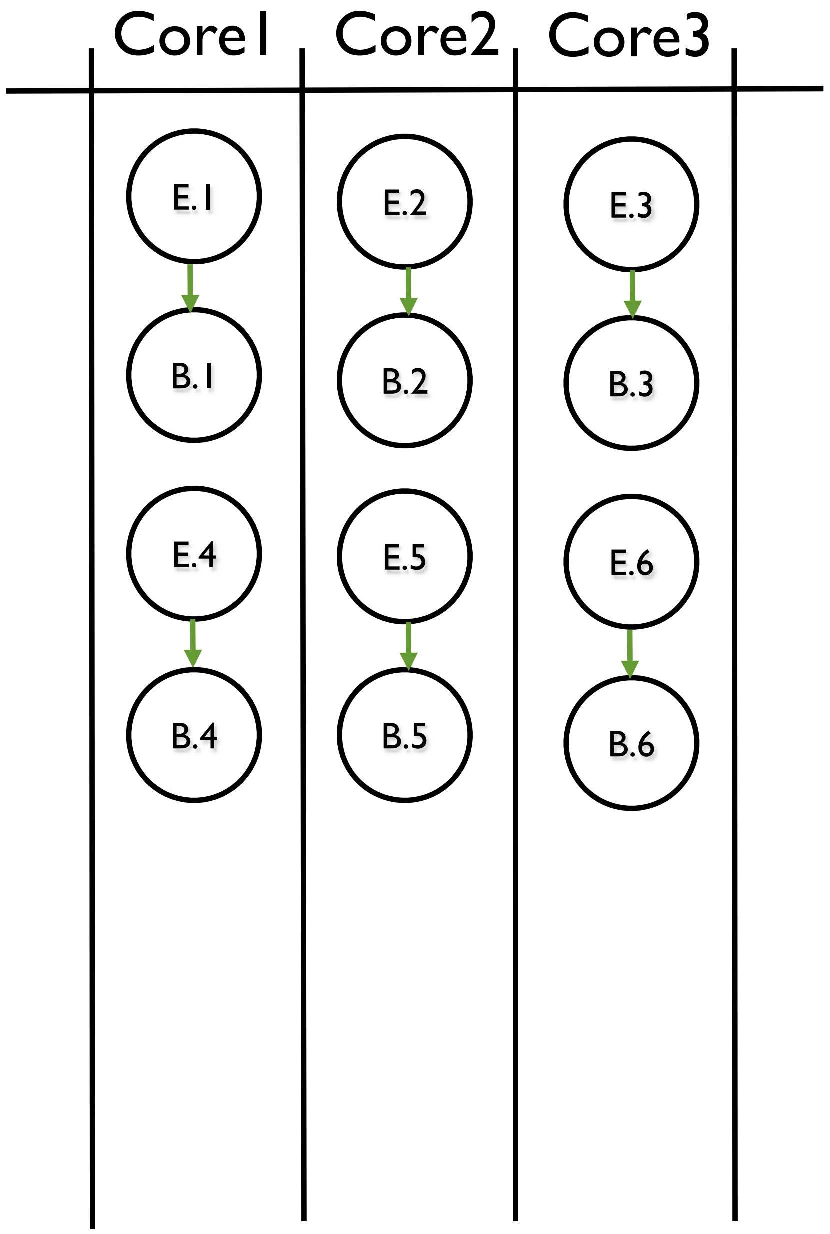
Sequential

```
    node = list -> head;
A:  while (node != NULL) {
D:      node = node -> next;
}
```



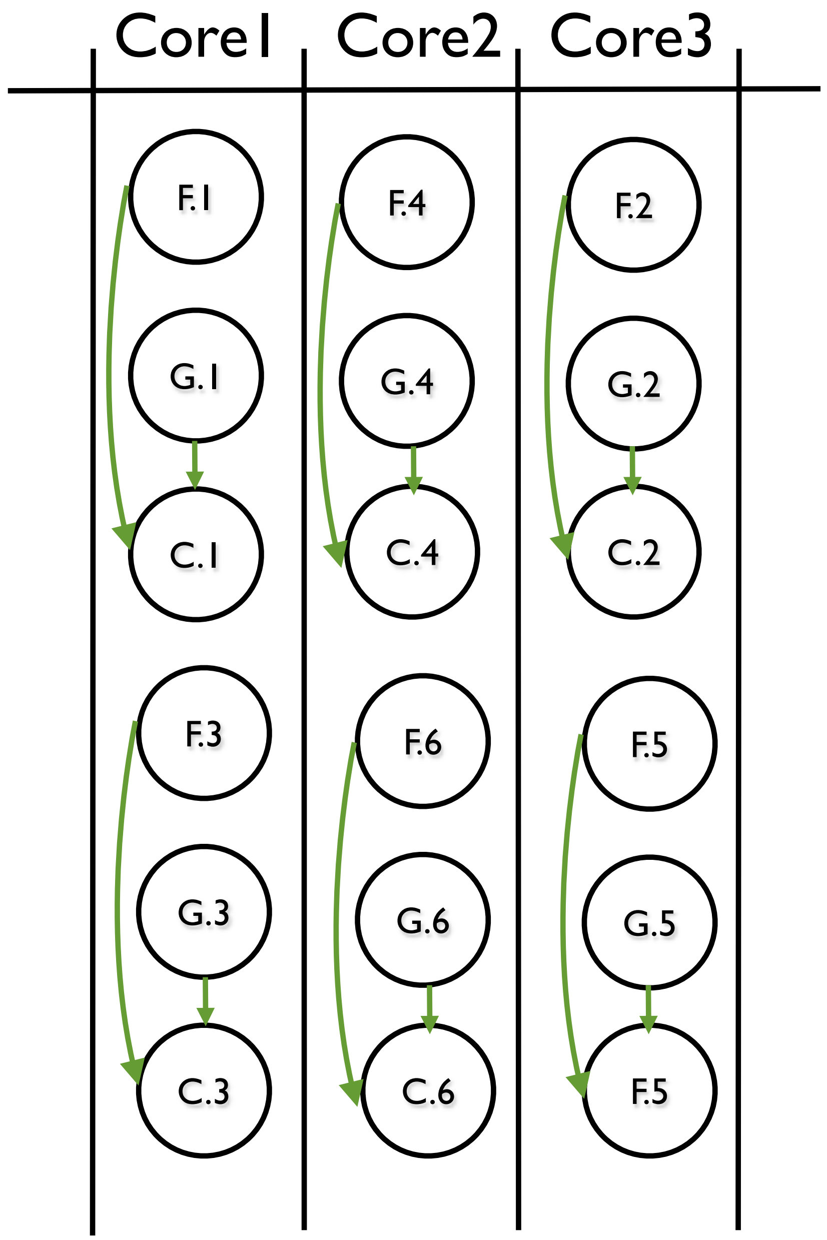
DOALL

```
    while (TRUE) {
E:    node = getNodeOrExit();
B:    index = calc
          (node -> data);
    }
```

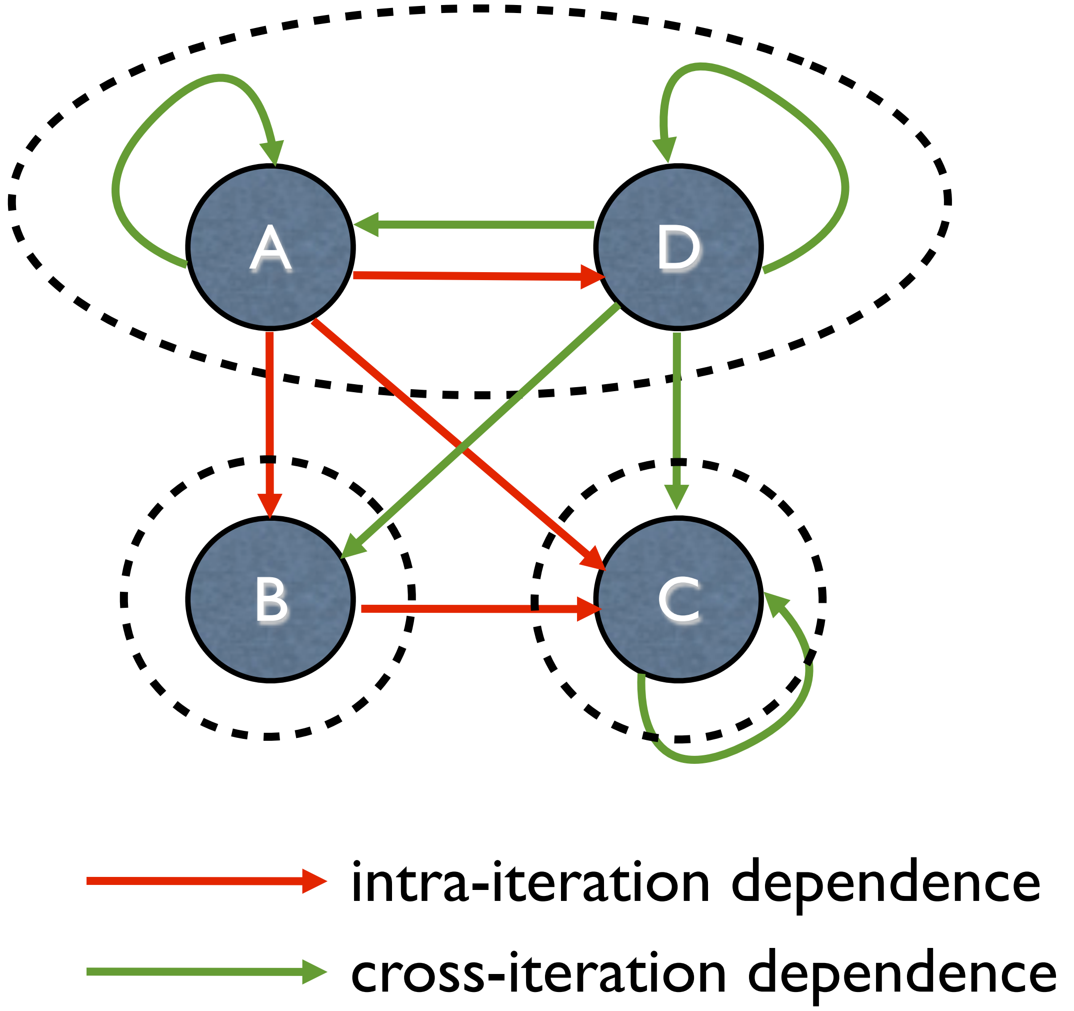


LOCALWRITE

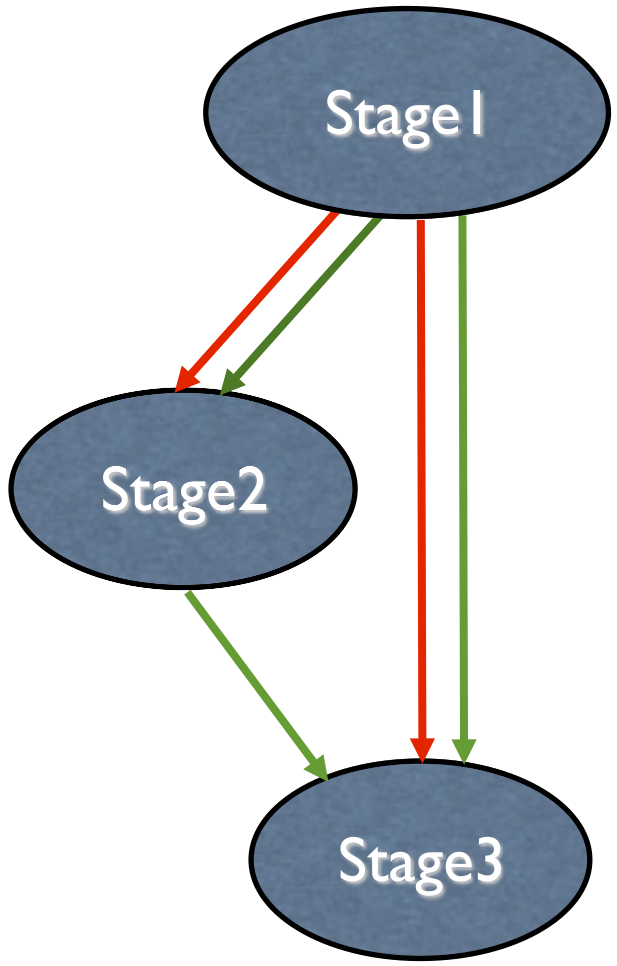
```
    while (TRUE) {
F:    node = getNodeOrExit();
G:    index = getIndex();
C:    density [index] = update_density
          (density [index], node -> data);
    }
```



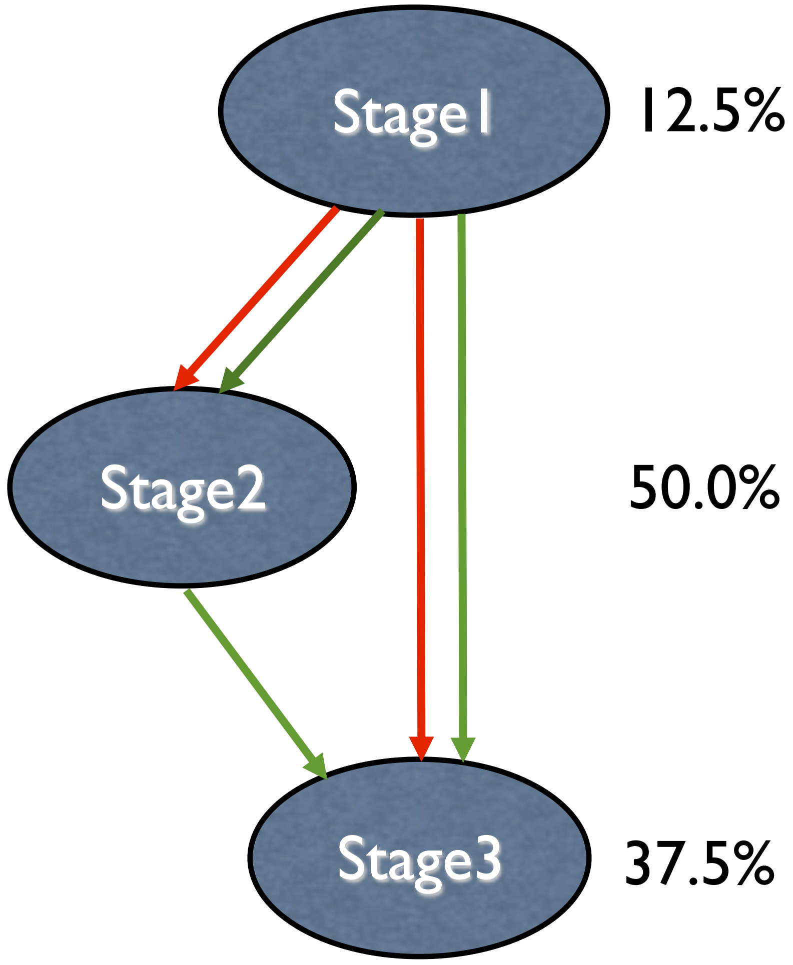
```
    node = list -> head;
A:  while (node != NULL) {
B:      index = calc (node -> data);
C:      density [index] = update_density
            (density [index], node -> data);
D:      node = node -> next;
    }
```



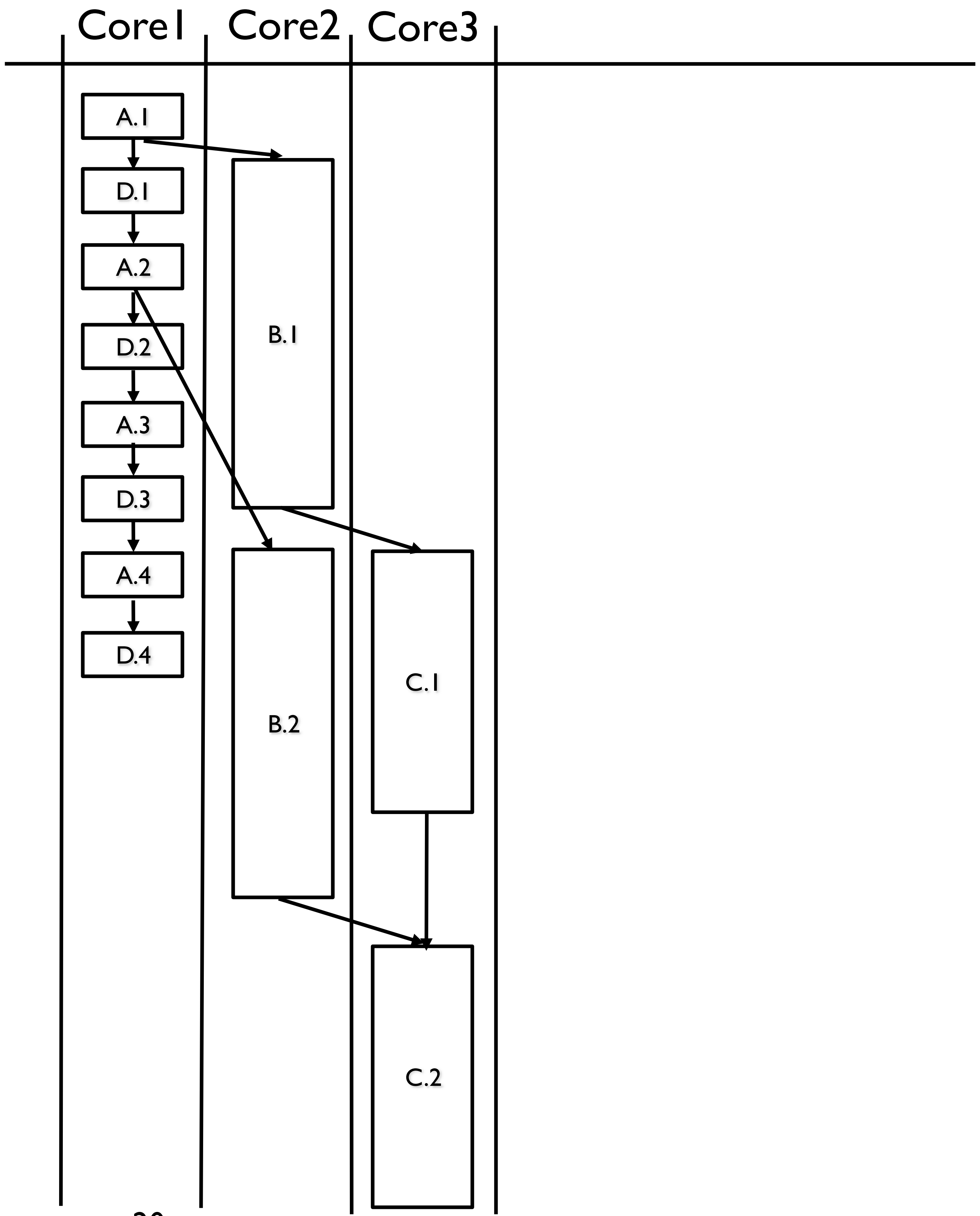
DSWP+



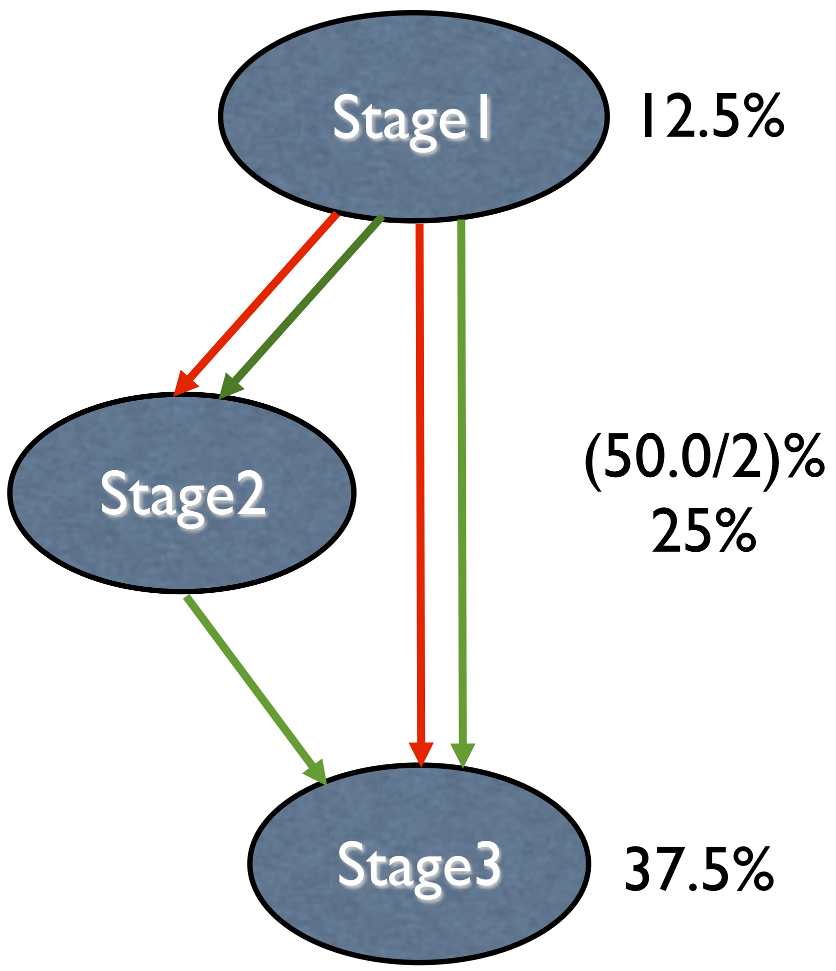
DSWP +

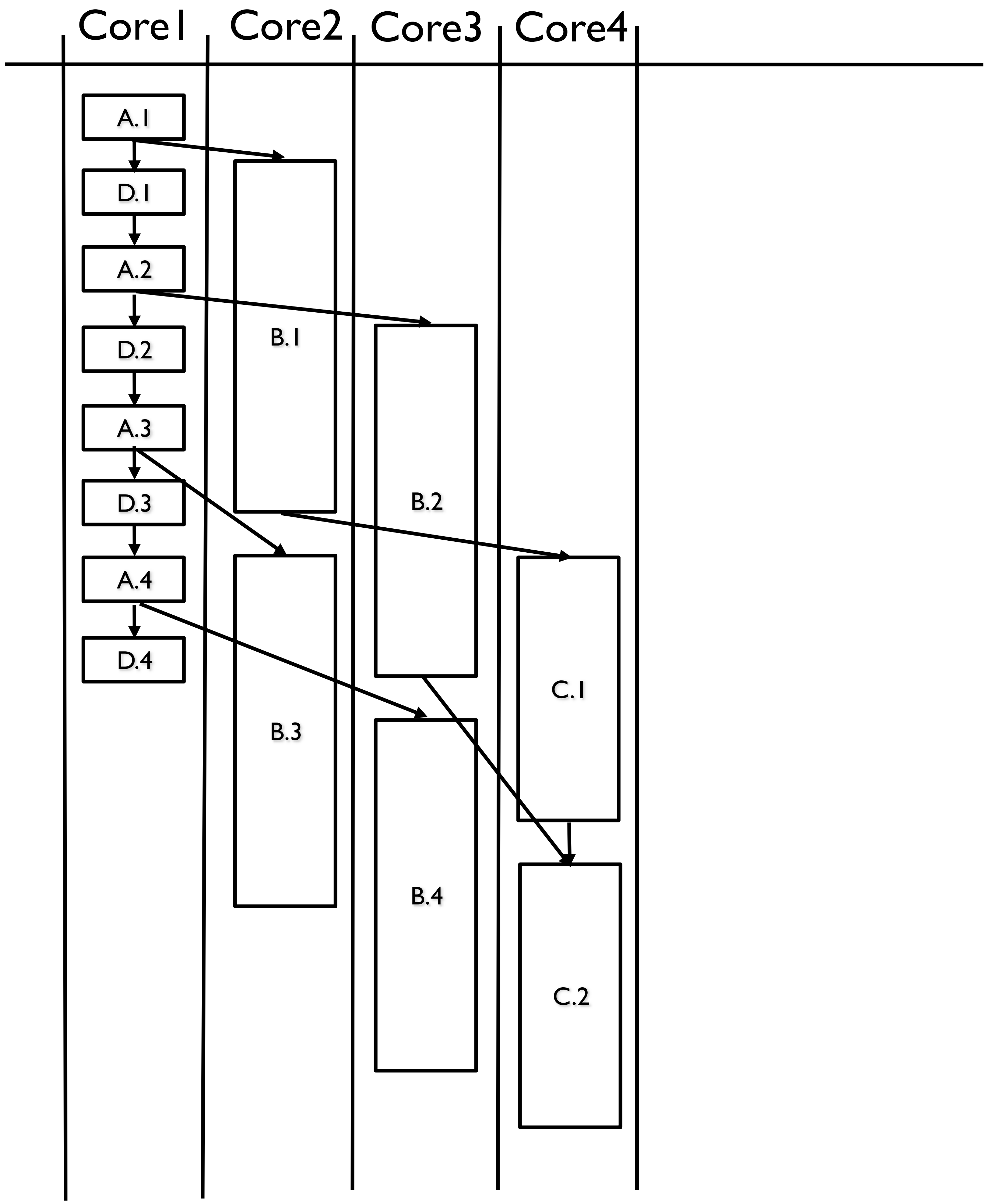


Max(12.5, 50.0, 37.5)
= 50.0 %
=> 2X (speedup)



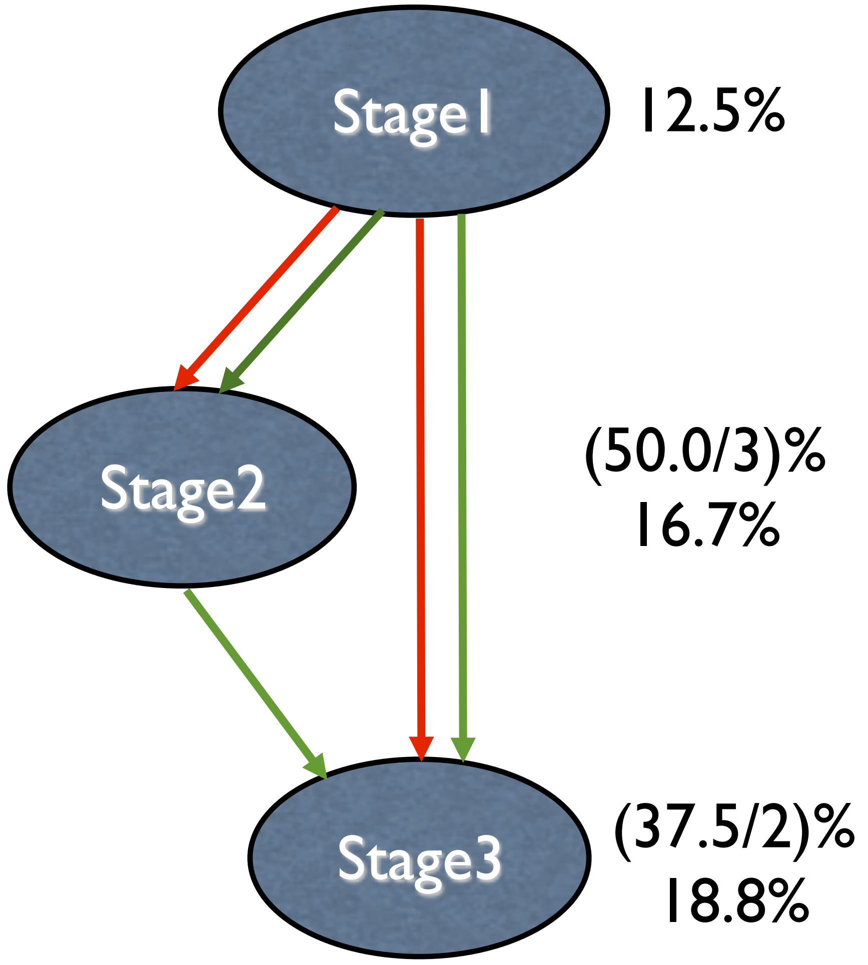
DSWP + DOALL

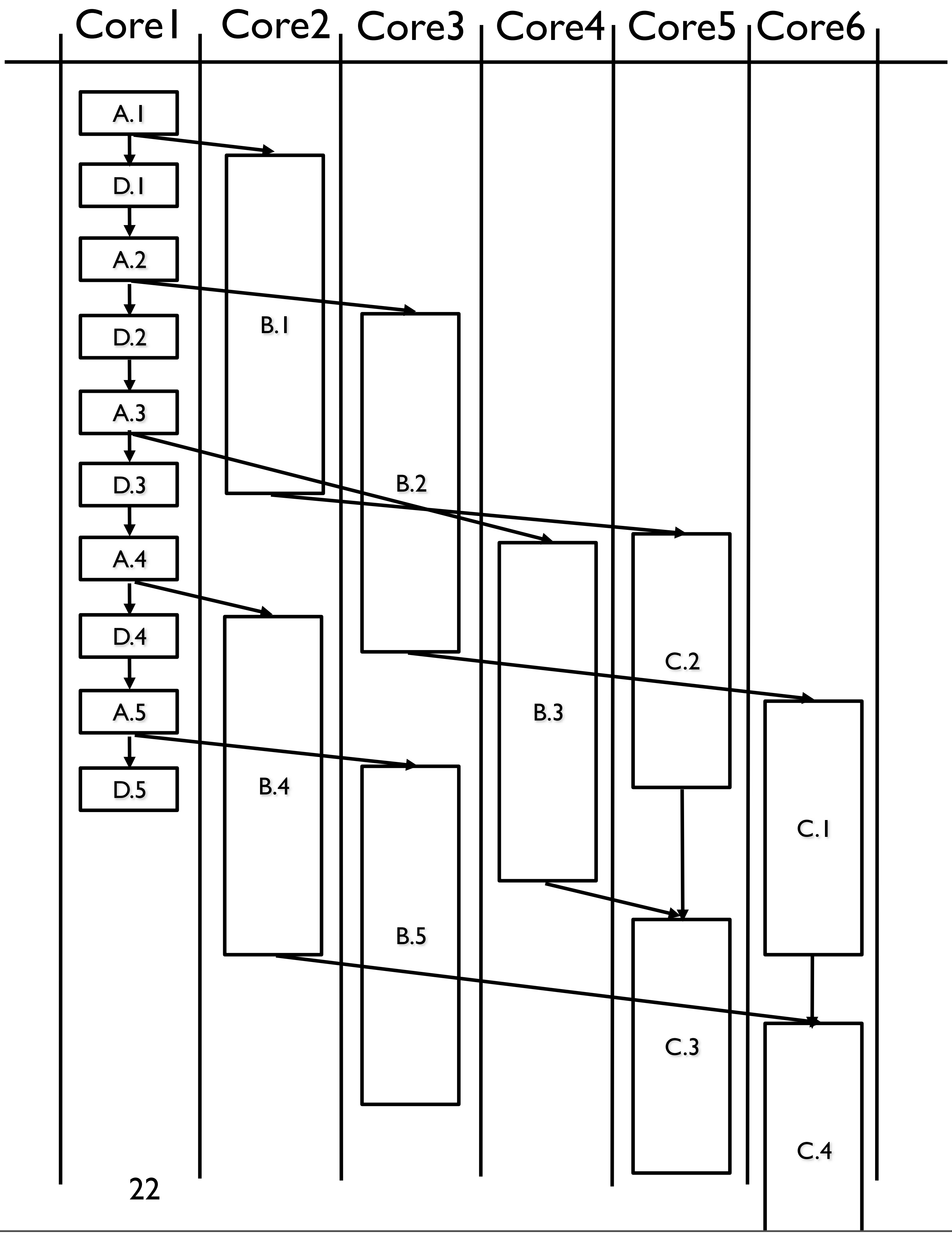




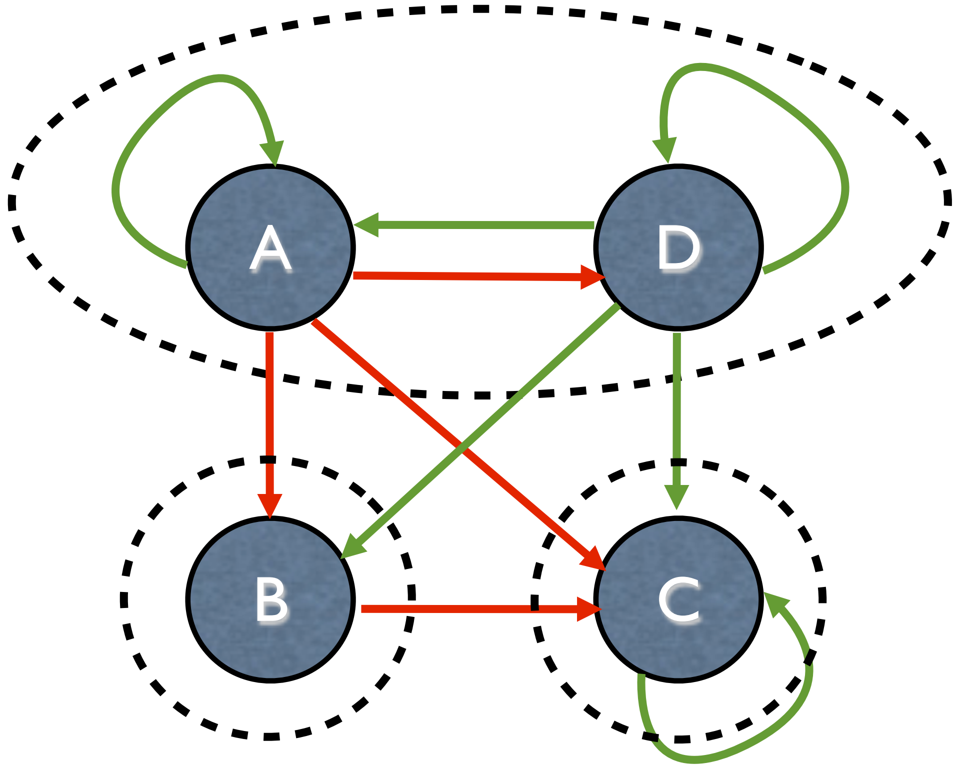
Max(12.5, 50.0/2, 37.5)
= 37.5 %
=> 2.7X (speedup)

DSWP + DOALL + LOCALWRITE





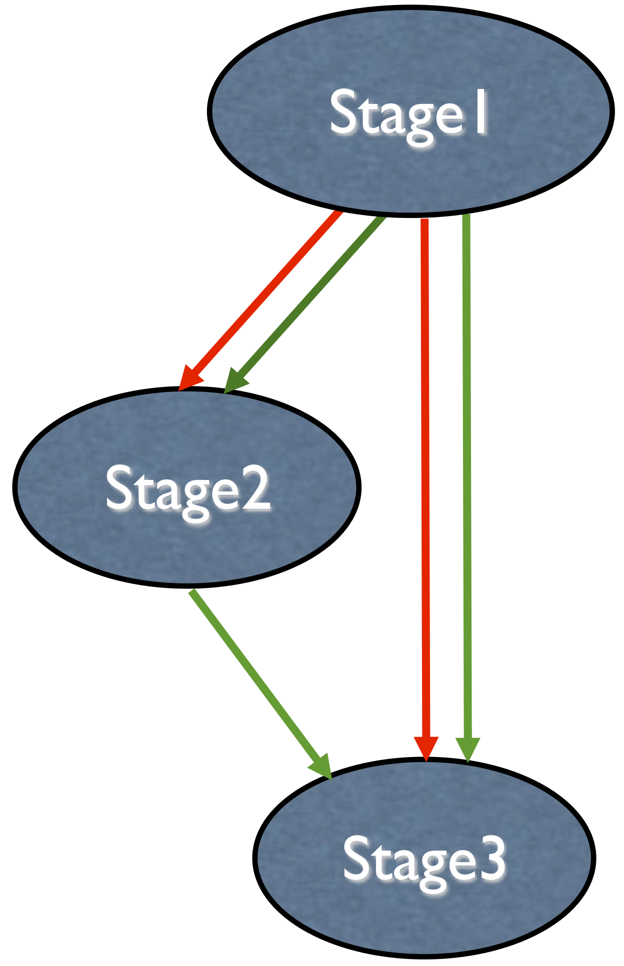
Max(12.5, 50.0/3, 37.5/2)
= 18.8 %
=> 5.3X (speedup)



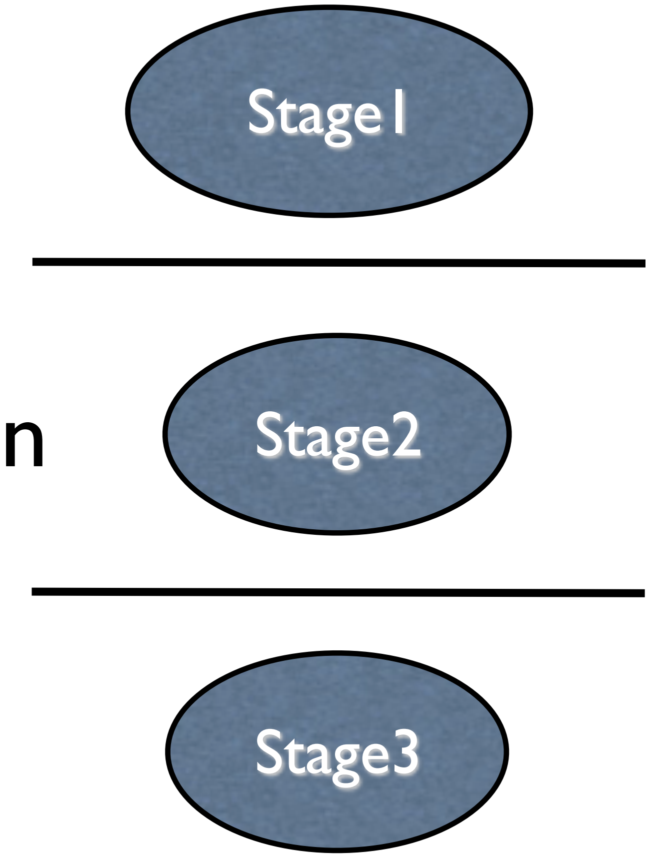
🟥 intra-iteration dependence

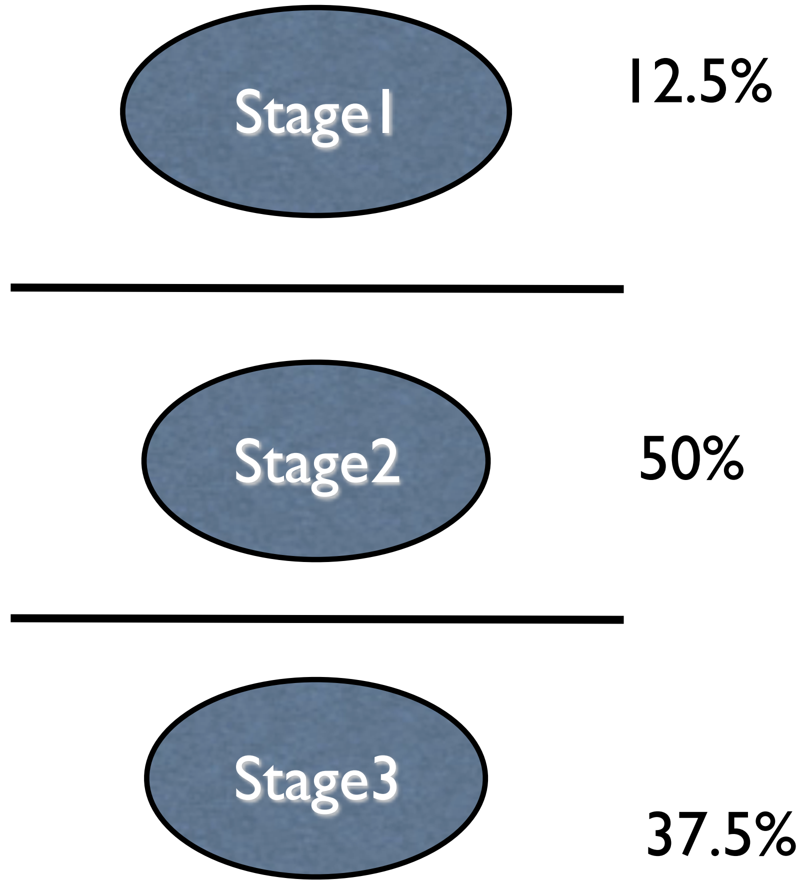
🟩 cross-iteration dependence

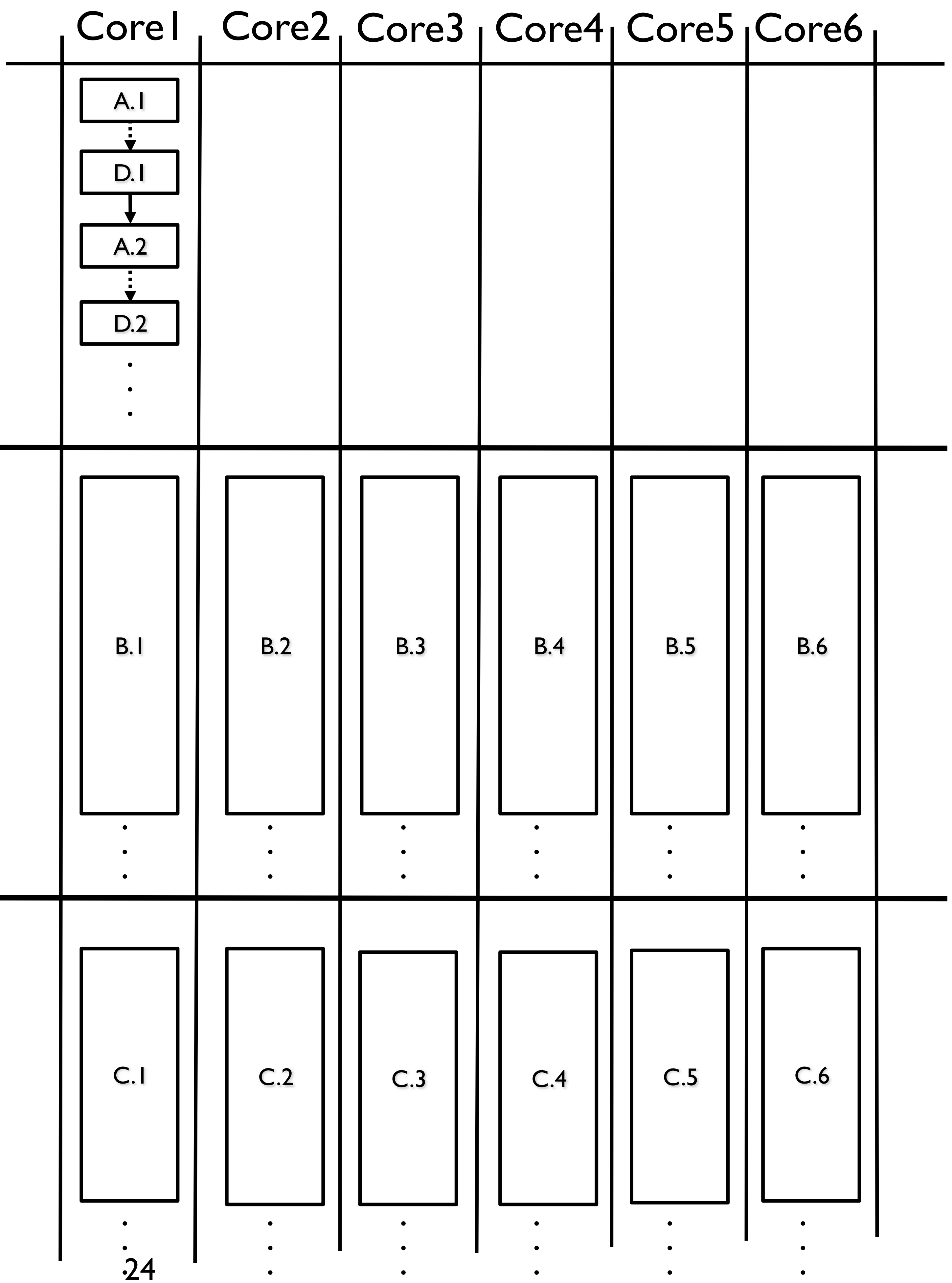
DSWP+



Loop Distribution

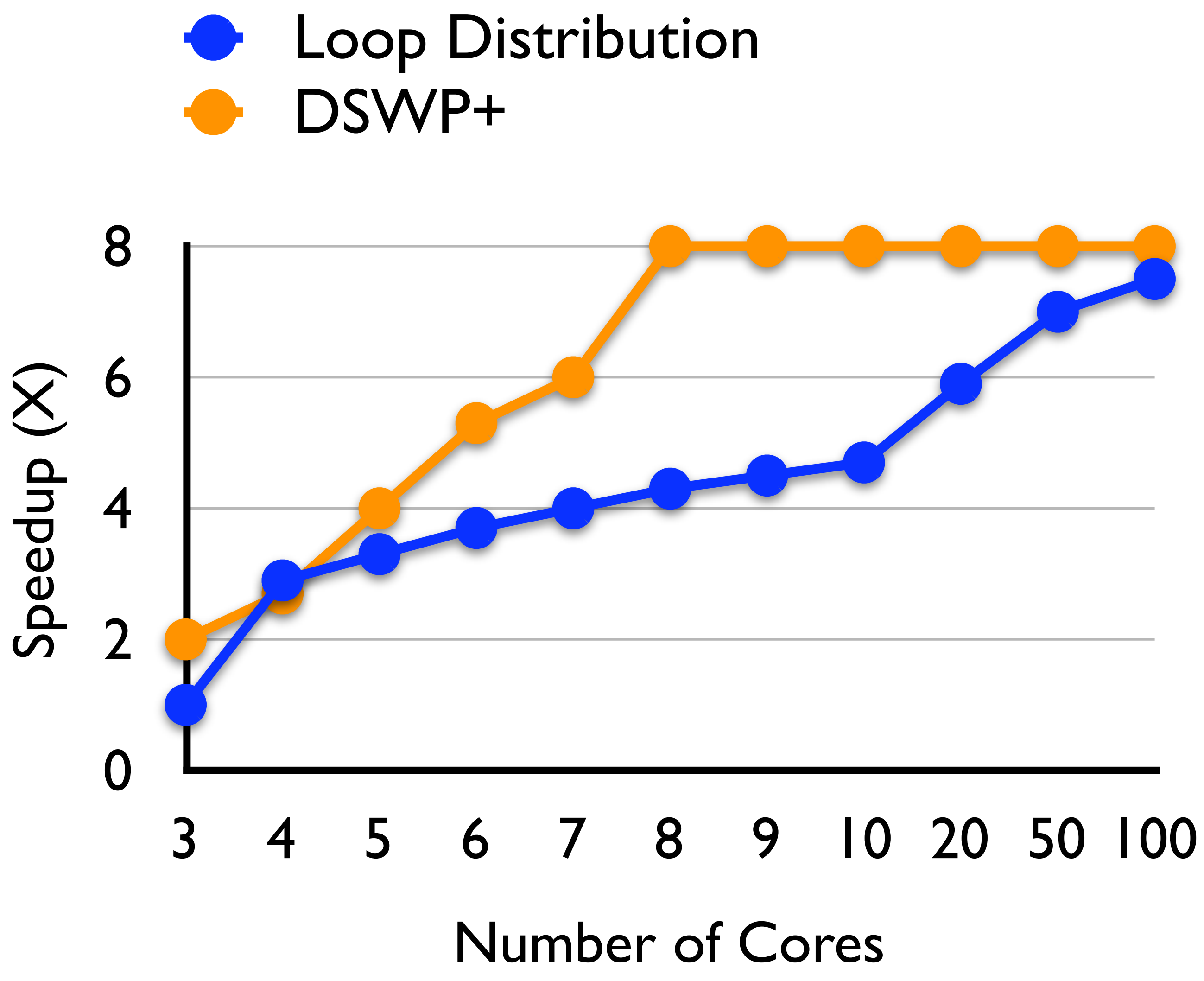




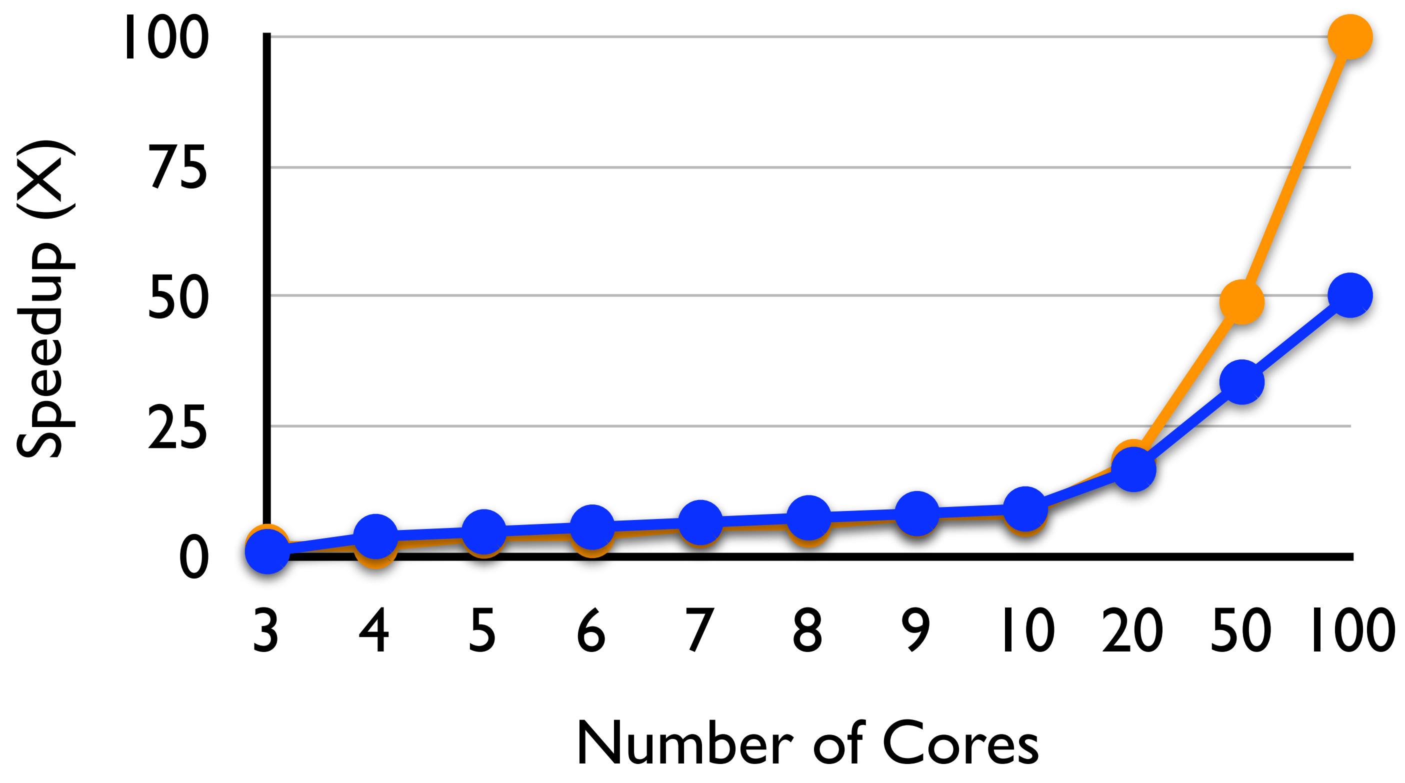


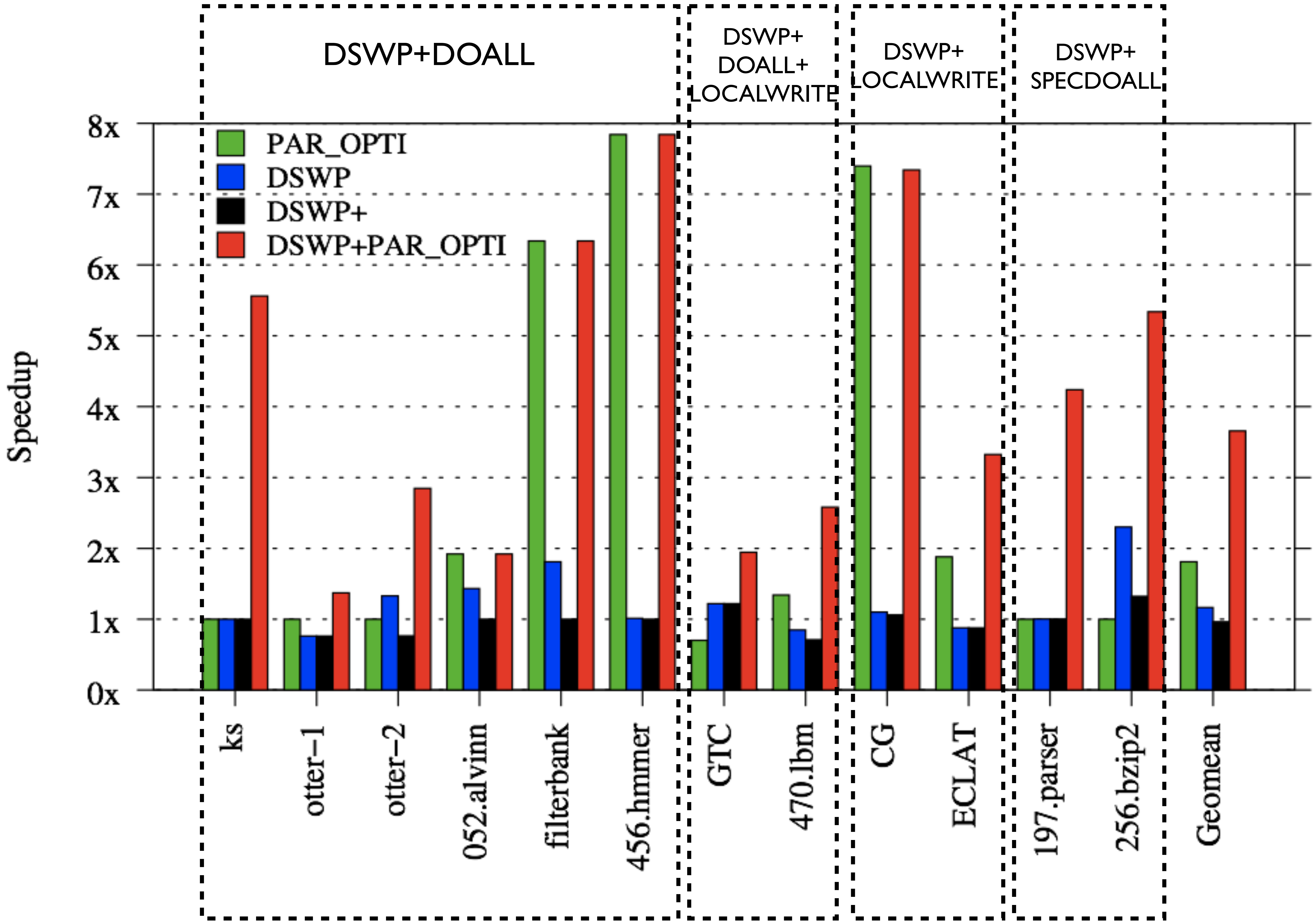
(12.5 + 50.0/6 + 37.5/6)
= 27.1%
=> 3.7X (speedup)

Stage Number	Execution Time (%)	Stage Type
1	**12.5**	Sequential
2	**50**	DOALL
3	**37.5**	LOCALWRITE



Stage Number	Execution Time (%)	Stage Type
1	**1**	Sequential
2	**50**	DOALL
3	**49**	LOCALWRITE




DSWP+DOALL
DSWP+ DOALL+ LOCALWRITE
DSWP+ LOCALWRITE
DSWP+ SPECDOALL
Speedup
8x
7x
6x
5x
4x
3x
2x
1x
0x
PAR_OPTI
DSWP
DSWP+
DSWP+PAR_OPTI
ks
otter–1
otter–2
052.alvinn
filterbank
456.hmmer
GTC
470.lbm
CG
ECLAT
197.parser
256.bzip2
Geomean

Questions?