Fiji Parallelization Cookbook

by

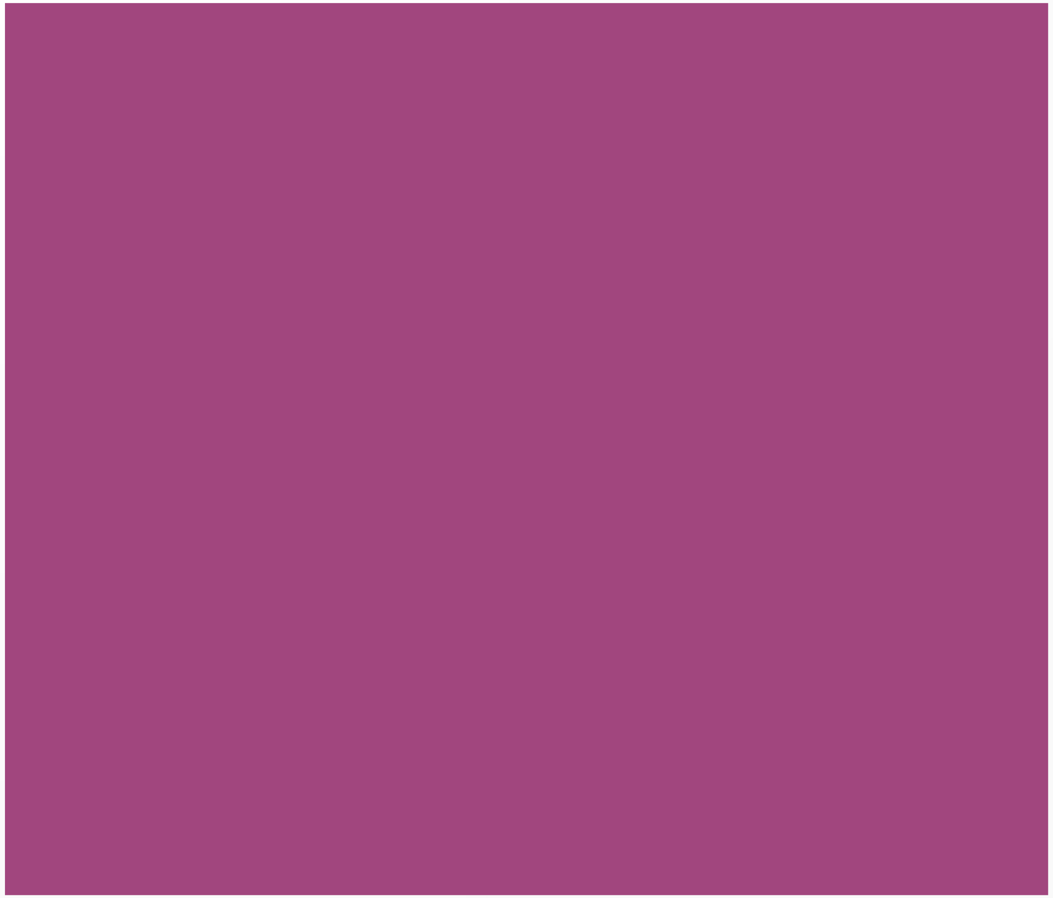
Stefanos, Vlado, Daniel & Michal

I have a huge image data...

- Abstract ***volume*** of *work* that needs to be done →

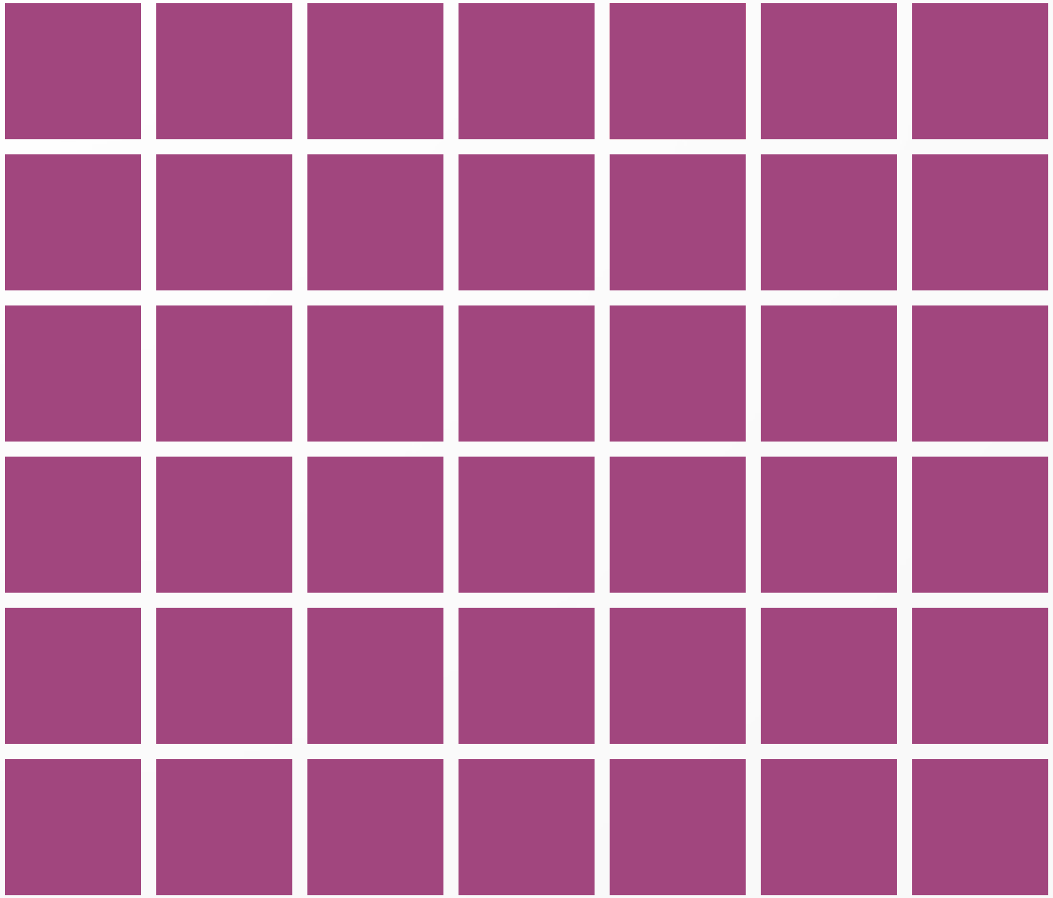
- Work = sequence of ImageJ commands

- Volume = set of tasks to work on



I have a huge image data...

- Abstract ***volume*** of *work* that needs to be done →
- Work = sequence of ImageJ commands
- Volume = set of tasks to work on



I have a huge image data...

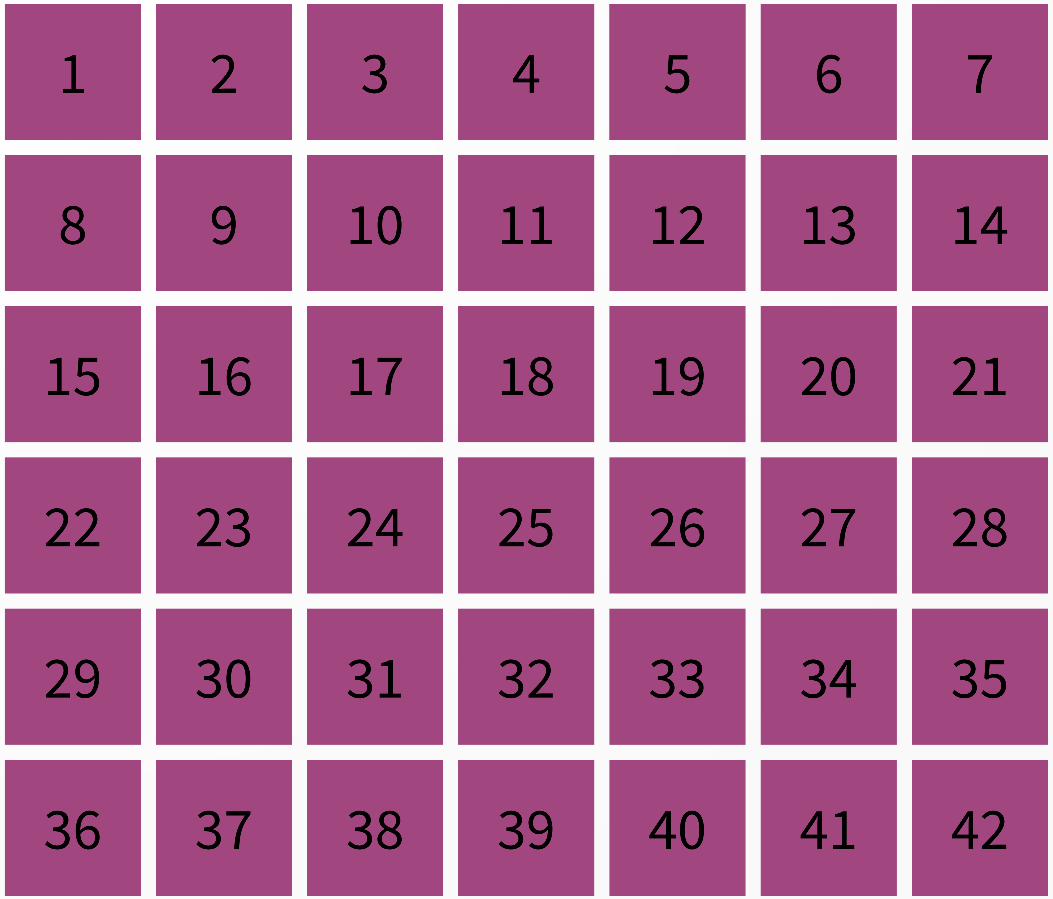
- Organize volume
  - Independent tasks
  - Number tasks with *i*

- Organize work
  - function *work*(*i*) {

    ...my code…

    }
  - Possibly also:

    load(*i*), *store*(*i*)



1	2	3	4	5	6	7
8	9	10	11	12	13	14
15	16	17	18	19	20	21
22	23	24	25	26	27	28
29	30	31	32	33	34	35
36	37	38	39	40	41	42

I process in sequence...

The famous pattern:

```
function work(i) {
    load(i);
    ...my (recorded) code…
    store(i);
}

for (i = 1; i <= 42; i++) {
    work(i);
}
```

1	2	3	4	5	6	7
8	9	10	11	12	13	14
15	16	17	18	19	20	21
22	23	24	25	26	27	28
29	30	31	32	33	34	35
36	37	38	39	40	41	42

I process in sequence...

```
for (i = 1; i <= 10; i++)
{ work(i); }
```

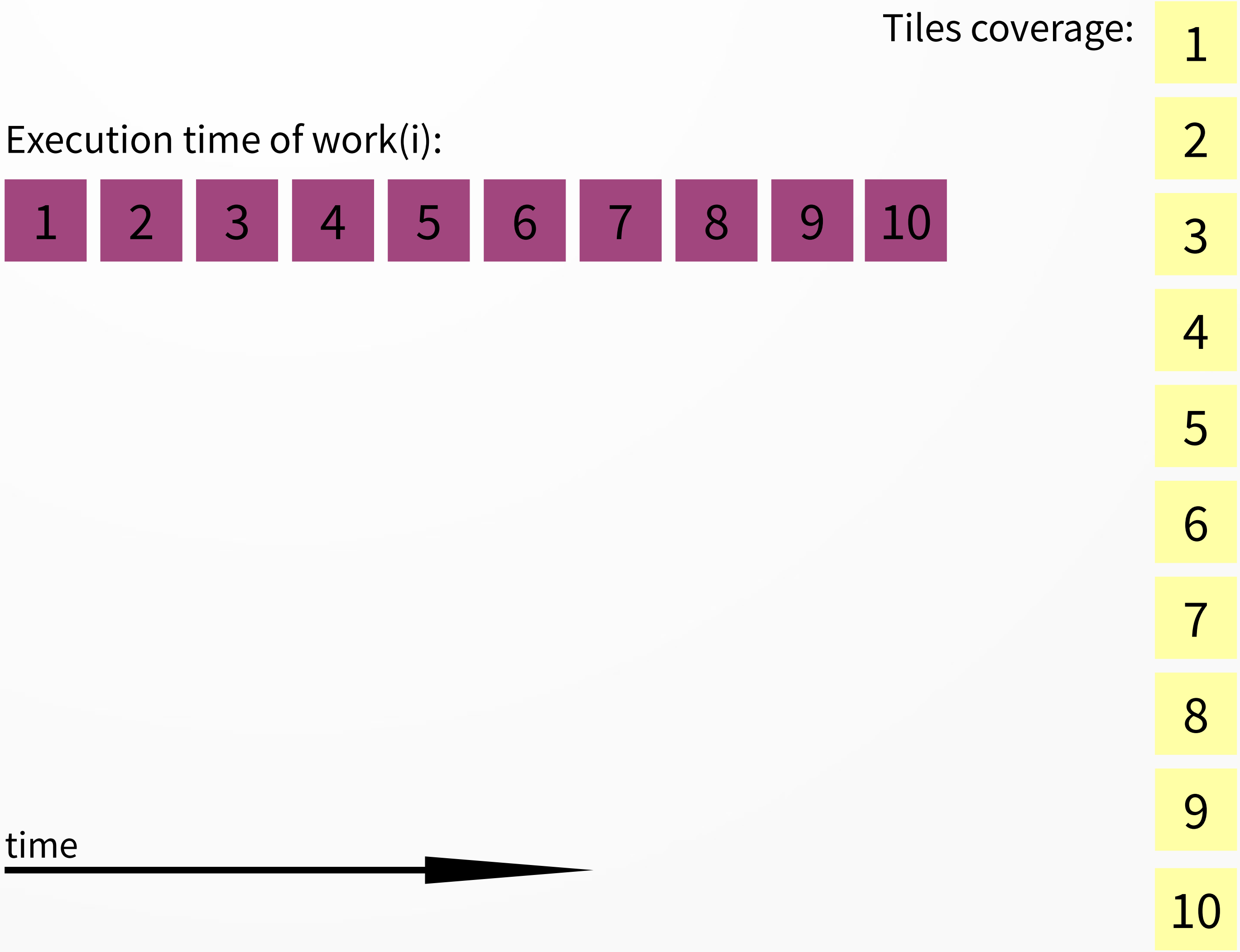
I process in sequence...

```
for (i = 1; i <= 10; i += 1)
{ work(i); }
```

I process in sequence...

The code runs in one instance:

```
for (i = 1; i <= 10; i += 1)
{ work(i); }
```



Execution time of work(i):

Tiles coverage:

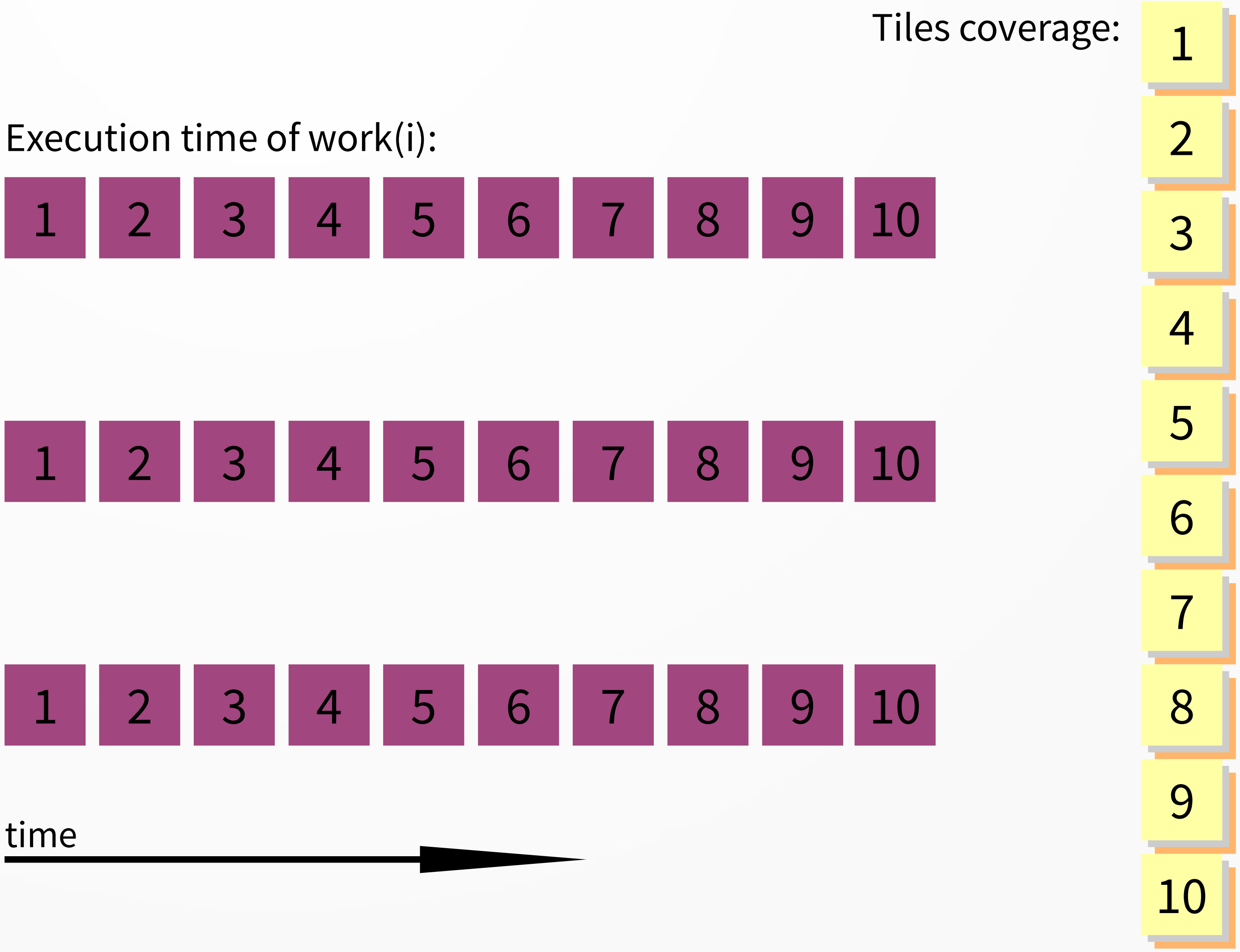
I process in parallel...

The code runs in three instances:

```
for (i = 1; i <= 10; i += 1)
{ work(i); }
```

```
for (i = 1; i <= 10; i += 1)
{ work(i); }
```

```
for (i = 1; i <= 10; i += 1)
{ work(i); }
```



Execution time of work(i):

Tiles coverage:

time

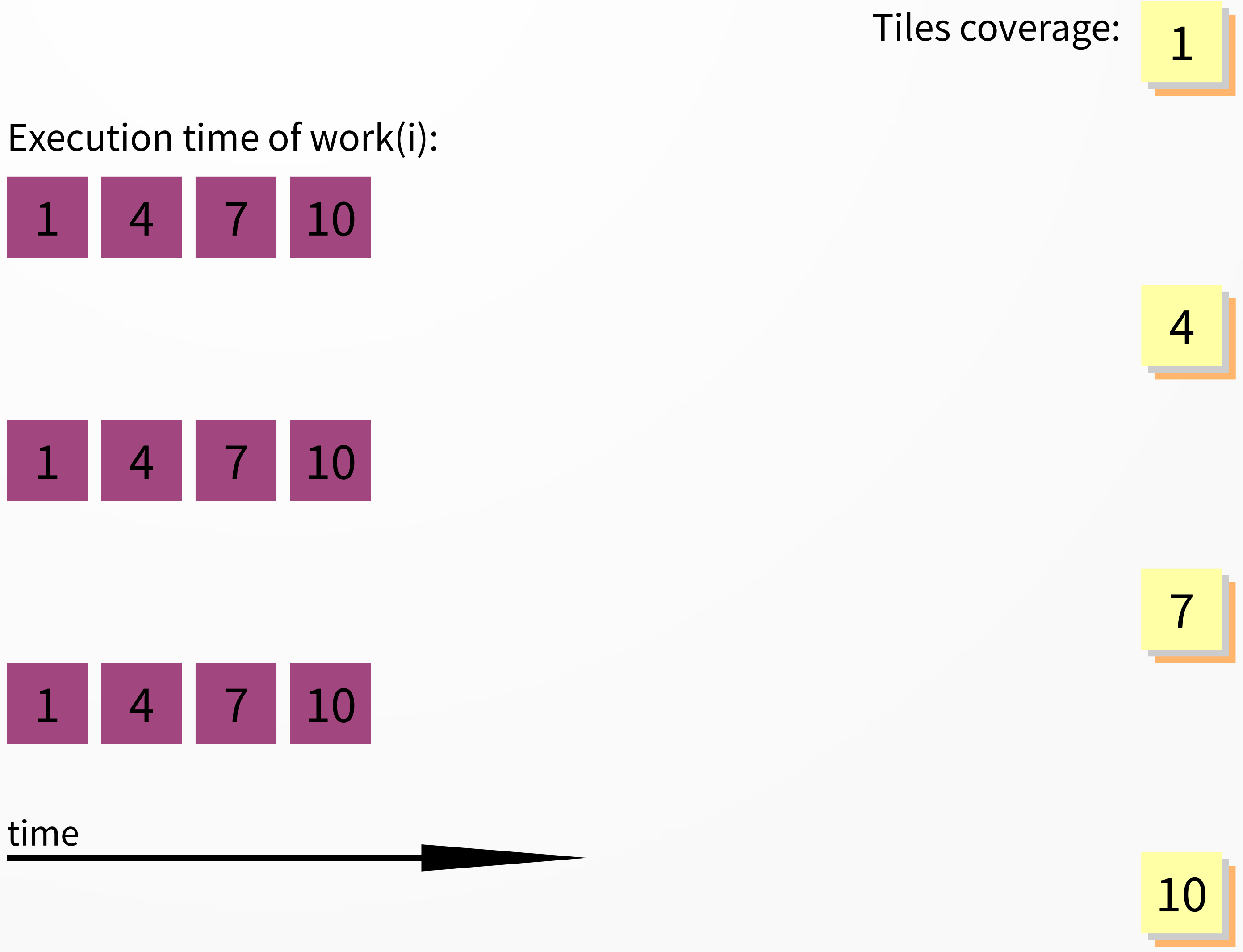
I process in parallel...

The code runs in three instances:

```
nCnt = parGetSize();            //is 3
for (i = 1; i <= 10; i += nCnt)
{ work(i); }
```

```
nCnt = parGetSize();            //is 3
for (i = 1; i <= 10; i += nCnt)
{ work(i); }
```

```
nCnt = parGetSize();            //is 3
for (i = 1; i <= 10; i += nCnt)
{ work(i); }
```



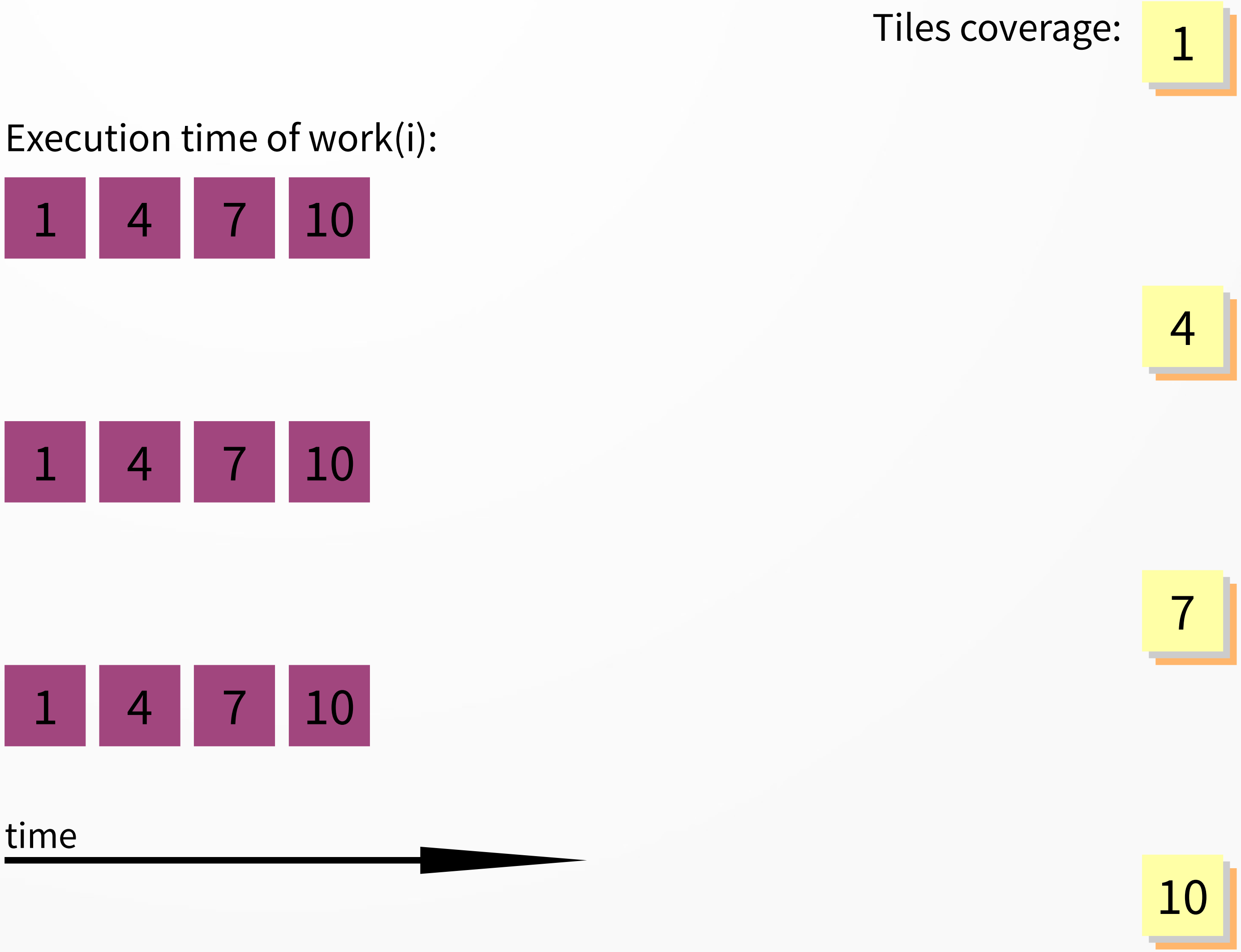
I process in parallel...

The code runs in three instances:

```
nCnt = parGetSize();              //is 3
for (i =     1; i <= 10; i += nCnt)
{ work(i); }
```

```
nCnt = parGetSize();              //is 3
for (i =     1; i <= 10; i += nCnt)
{ work(i); }
```

```
nCnt = parGetSize();              //is 3
for (i =     1; i <= 10; i += nCnt)
{ work(i); }
```



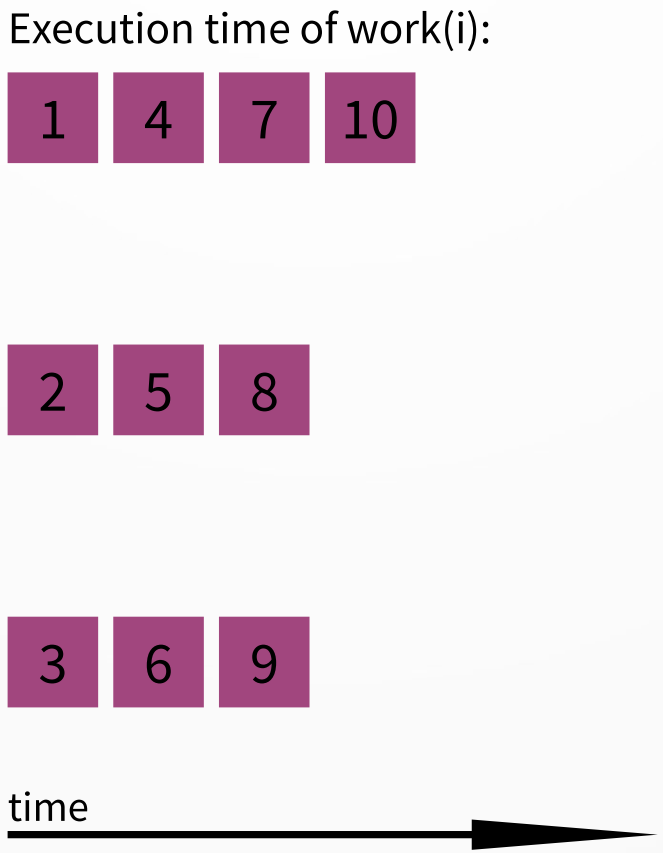
I process in parallel...

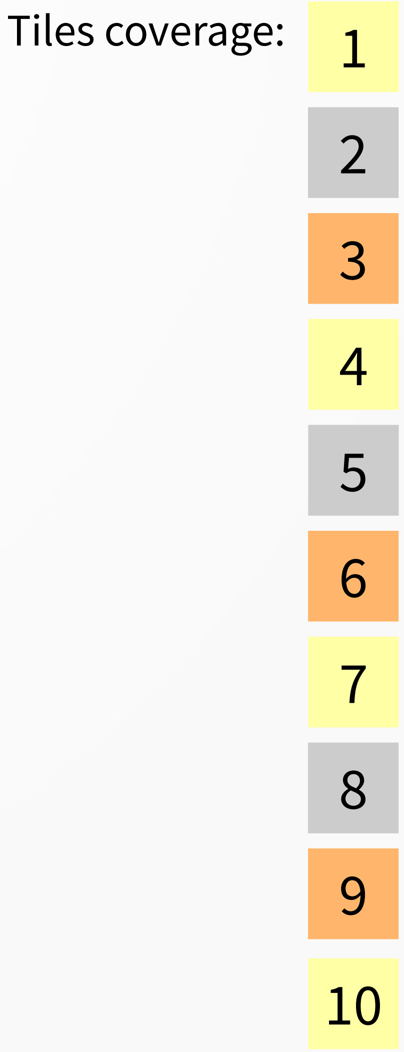
The code runs in three instances:

```
nID = parGetRank()+1;           //is 1
nCnt = parGetSize();            //is 3
for (i = nID; i <= 10; i += nCnt)
{ work(i); }
```

```
nID = parGetRank()+1;           //is 2
nCnt = parGetSize();            //is 3
for (i = nID; i <= 10; i += nCnt)
{ work(i); }
```

```
nID = parGetRank()+1;           //is 3
nCnt = parGetSize();            //is 3
for (i = nID; i <= 10; i += nCnt)
{ work(i); }
```

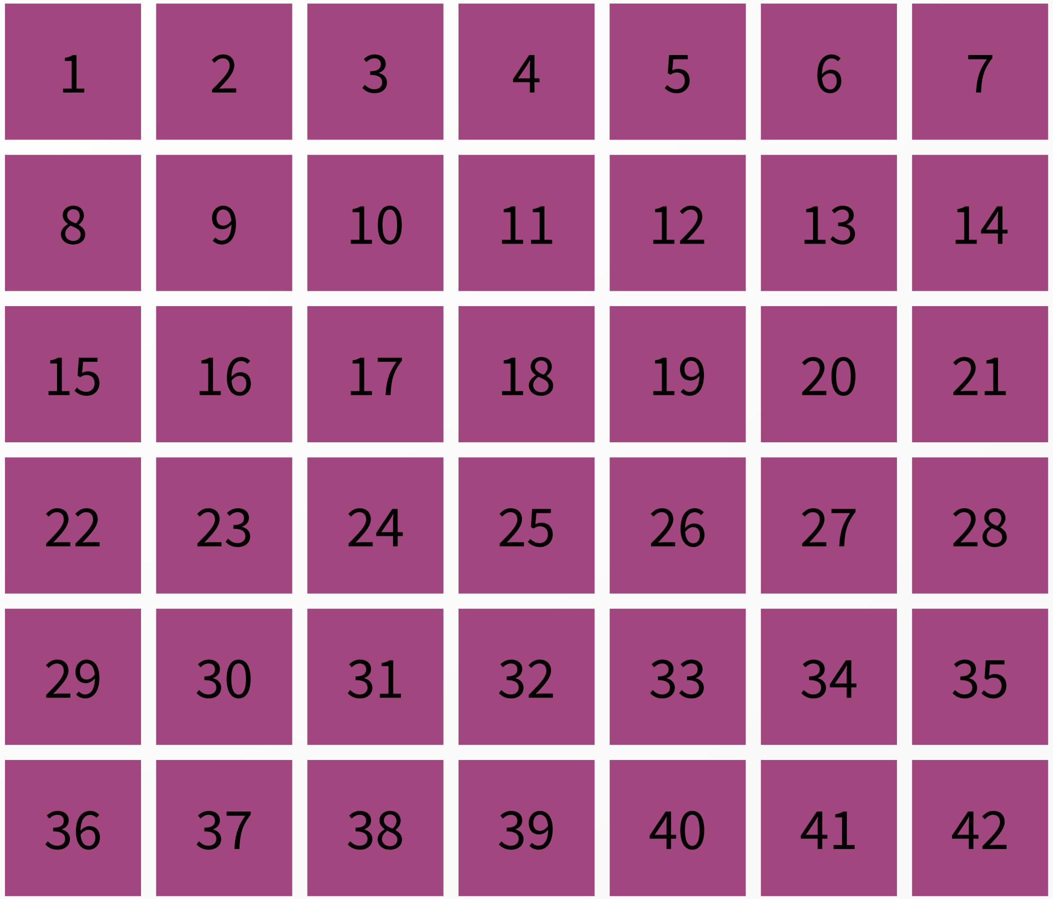

Execution time of work(i):



I process in sequence...

From basic serial pattern:

```
function work(i) {
    load(i);
    ...my code…
    store(i);
}


for (i = 1; i <= 42; i++) {
    work(i);
}
```

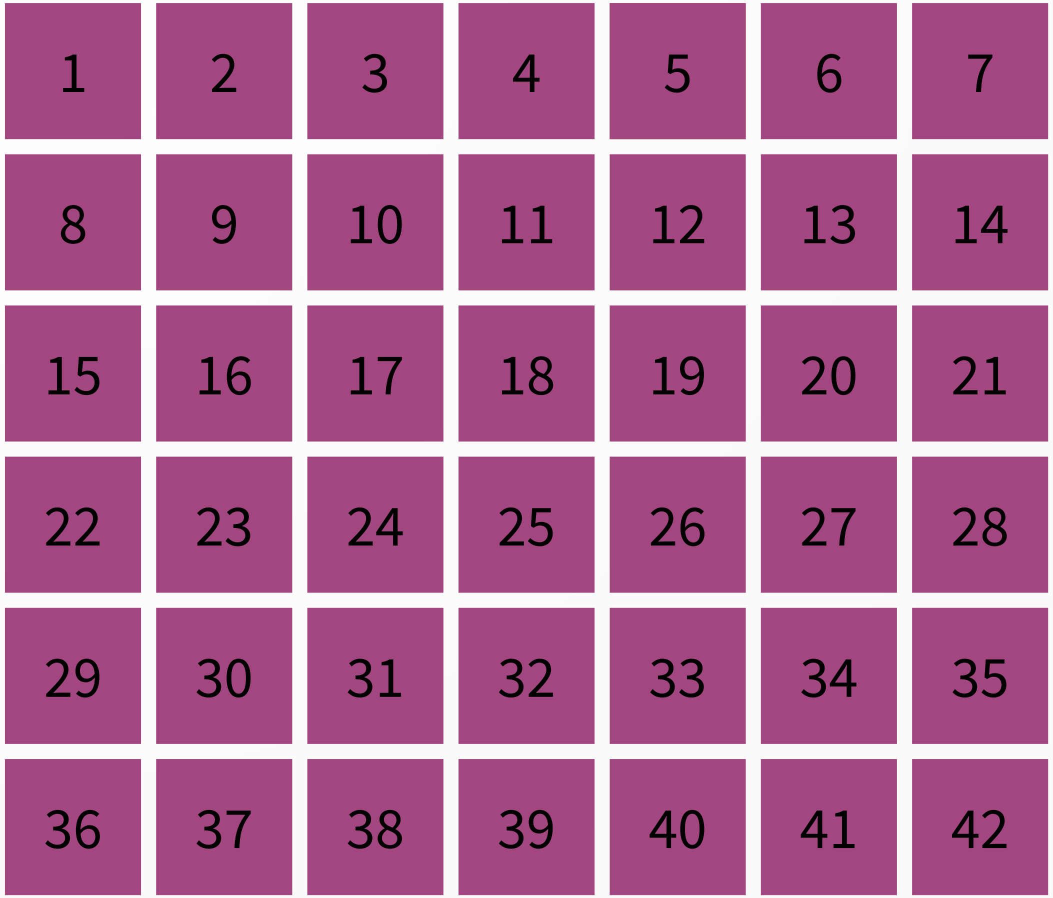


1	2	3	4	5	6	7
8	9	10	11	12	13	14
15	16	17	18	19	20	21
22	23	24	25	26	27	28
29	30	31	32	33	34	35
36	37	38	39	40	41	42

I process in parallel...

To easy parallel pattern:

```
function work(i) {
    load(i);
    ...my code…
    store(i);
}

parInit();
for (i=parGetRank(); i <= 42; i+=parGetSize()) {
    work(i);
}
parFinalize();
```

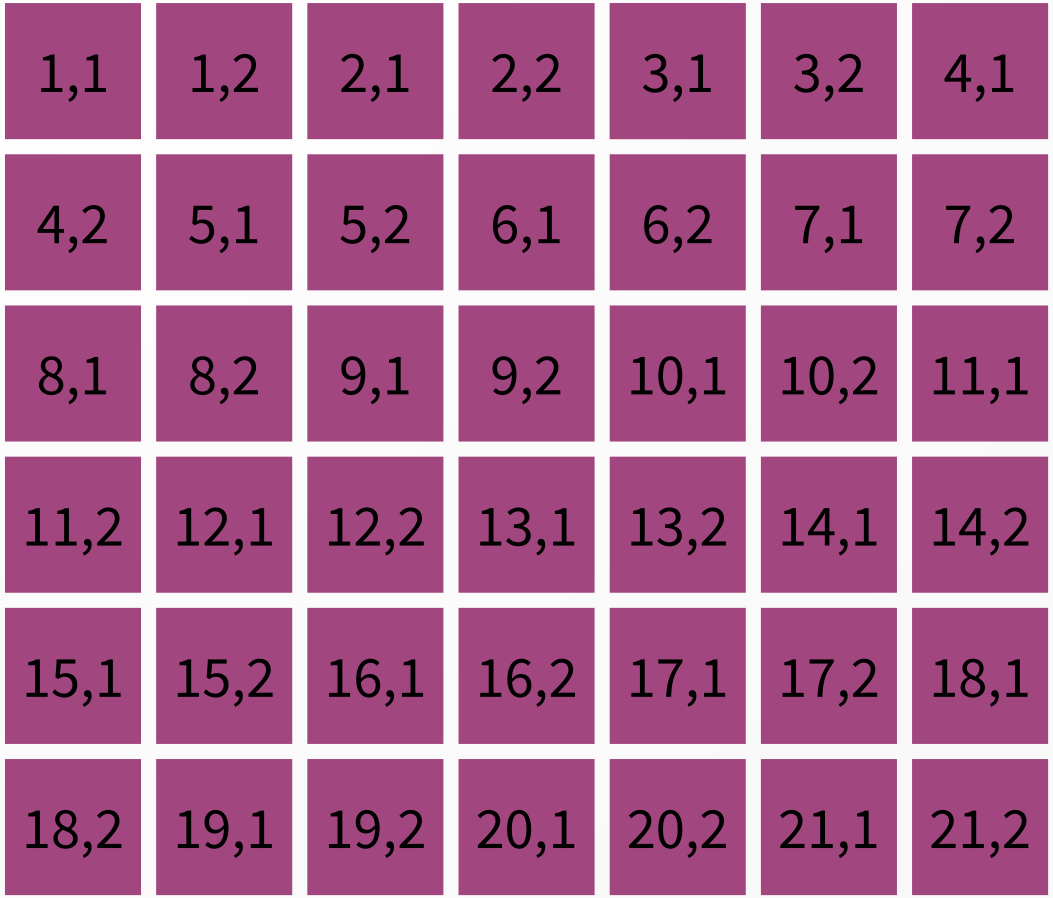


1	2	3	4	5	6	7
8	9	10	11	12	13	14
15	16	17	18	19	20	21
22	23	24	25	26	27	28
29	30	31	32	33	34	35
36	37	38	39	40	41	42

I process in parallel...

To advanced parallel pattern:

```
function workCh1(i) {...my code…}
function workCh2(i) {...my code…}

parInit();
for (i=parGetRank(); i <= 42; i+=parGetSize()) {
    if (i % 2 == 0) {
        workCh1( floor(i/2) );
    } else {
        workCh2( floor(i/2) );
    }
}
parFinalize();
```



1,1	1,2	2,1	2,2	3,1	3,2	4,1
4,2	5,1	5,2	6,1	6,2	7,1	7,2
8,1	8,2	9,1	9,2	10,1	10,2	11,1
11,2	12,1	12,2	13,1	13,2	14,1	14,2
15,1	15,2	16,1	16,2	17,1	17,2	18,1
18,2	19,1	19,2	20,1	20,2	21,1	21,2

I process in parallel...

Silent parallel pattern:

```
parInit();

function work(i) {
    load(i);
    ...my code…
    store(i);
}


for (i=parGetRank(); i <= 42; i+=parGetSize()) {
    work(i);
}
parFinalize();
```

Verbose parallel pattern:

```
parInit();



function work(i) {
    load(i);


    ...my code…


    store(i);

}

for (i=parGetRank(); i <= 42; i+=parGetSize()) {
    work(i);
}
parFinalize();
```

I process in parallel...

Silent parallel pattern:

```
parInit();

function work(i) {
    load(i);
    ...my code…
    store(i);
}


for (i=parGetRank(); i <= 42; i+=parGetSize()) {
    work(i);
}
parFinalize();
```

Verbose parallel pattern:

```
parInit();
for (i=parGetRank(); i <= 42; i+=parGetSize()) {
    parAddTask("Img:  "+i);        // work segments (tasks) count from 0
}


function work(i) {
    load(i);


    ...my code…

    store(i);

}

for (i=parGetRank(); i <= 42; i+=parGetSize()) {
    work(i);
}
parFinalize();
```

I process in parallel...

Silent parallel pattern:

```
parInit();

function work(i) {
    load(i);
    ...my code…
    store(i);
}


for (i=parGetRank(); i <= 42; i+=parGetSize()) {
    work(i);
}
parFinalize();
```

Verbose parallel pattern:

```
parInit();
for (i=parGetRank(); i <= 42; i+=parGetSize()) {
    parAddTask(“Img:  “+i);         // work segments (tasks) count from 0
}
parReportTasks();                   // submit created work segments

function work(i) {
    load(i);


    ...my code…

    store(i);

}

for (i=parGetRank(); i <= 42; i+=parGetSize()) {
    work(i);
}
parFinalize();
```

I process in parallel...

Silent parallel pattern:

```
parInit();

function work(i) {
    load(i);
    ...my code…
    store(i);
}

for (i=parGetRank(); i <= 42; i+=parGetSize()) {
    work(i);
}
parFinalize();
```

Verbose parallel pattern:

```
parInit();
for (i=parGetRank(); i <= 42; i+=parGetSize()) {
   parAddTask(“Img:  “+i);          // work segments (tasks) count from 0
}
parReportTasks();                   // submit created work segments

function work(i) {
   load(i);
   taskNo = floor(i / parGetSize());    // =0,1,2,3,4,5,...

   ...my code…

   store(i);

}

for (i=parGetRank(); i <= 42; i+=parGetSize()) {
   work(i);
}
parFinalize();
```

I process in parallel...

Silent parallel pattern:

```
parInit();

function work(i) {
    load(i);
    ...my code…
    store(i);
}


for (i=parGetRank(); i <= 42; i+=parGetSize()) {
    work(i);
}
parFinalize();
```

Verbose parallel pattern:

```
parInit();
for (i=parGetRank(); i <= 42; i+=parGetSize()) {
    parAddTask(“Img:  “+i);        // work segments (tasks) count from 0
}
parReportTasks();                  // submit created work segments

function work(i) {
    load(i);
    taskNo = floor(i / parGetSize());   // =0,1,2,3,4,5,...
    parReportProgress(taskNo, 20);  // advanced in work segment
    ...my code…

    store(i);

}

for (i=parGetRank(); i <= 42; i+=parGetSize()) {
    work(i);
}
parFinalize();
```

I process in parallel...

Silent parallel pattern:

```
parInit();

function work(i) {
    load(i);
    ...my code…
    store(i);
}

for (i=parGetRank(); i <= 42; i+=parGetSize()) {
    work(i);
}
parFinalize();
```

Verbose parallel pattern:

```
parInit();
for (i=parGetRank(); i <= 42; i+=parGetSize()) {
    parAddTask(“Img:  “+i);        // work segments (tasks) count from 0
}
parReportTasks();                  // submit created work segments

function work(i) {
    load(i);
    taskNo = floor(i / parGetSize());    // =0,1,2,3,4,5,...
    parReportProgress(taskNo, 20);  // advanced in work segment
    ...my code…
    parReportProgress(taskNo, 80);
    store(i);
    parReportProgress(taskNo, 100);
}

for (i=parGetRank(); i <= 42; i+=parGetSize()) {
    work(i);
}
parFinalize();
```

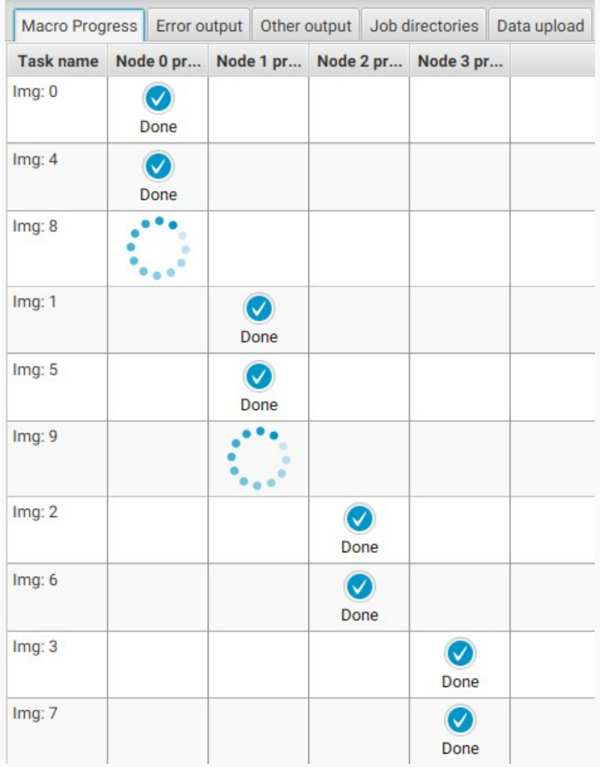
I process in parallel… yay!

Original serial pattern:

```
function work(i) {
    load(i);
    ...my (recorded) code…
    store(i);
}


for (i = 1; i <= 42; i++) {
    work(i);
}
```

Verbose parallel pattern:

```
parInit();
for (i=parGetRank(); i <= 42; i+=parGetSize()) {
    parAddTask(“Img:  “+i);        // work segments (tasks) count from 0
}
parReportTasks();                  // submit created work segments

function work(i) {
    load(i);
    taskNo = floor(i / parGetSize());    // =0,1,2,3,4,5,...
    parReportProgress(taskNo, 20);   // advancement
    ...my (recorded) code…
    parReportProgress(taskNo, 80);
    store(i);
    parReportProgress(taskNo, 100);
}

for (i=parGetRank(); i <= 42; i+=parGetSize()) {
    work(i);
}
parFinalize();
```



Macro Progress | Error output | Other output | Job directories | Data upload

Task name	Node 0 pr...	Node 1 pr...	Node 2 pr...	Node 3 pr...
Img: 0	✓ Done			
Img: 4	✓ Done			
Img: 8	⟳			
Img: 1		✓ Done		
Img: 5		✓ Done		
Img: 9		⟳		
Img: 2			✓ Done	
Img: 6			✓ Done	
Img: 3				✓ Done
Img: 7				✓ Done

I process in parallel… yay!

Original serial pattern:

```
function work(i) {
    load(i);
    ...my (recorded) code…
    store(i);
}

for (i = 1; i <= 42; i++) {
    work(i);
}
```

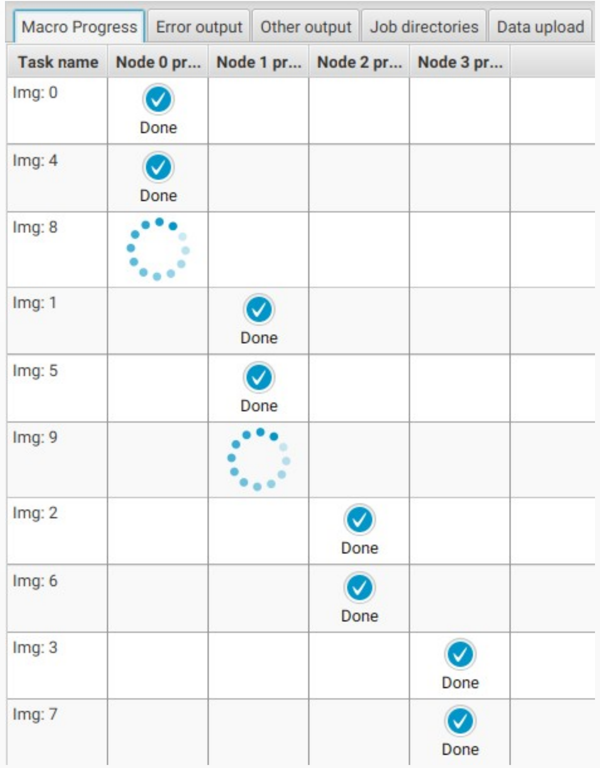
Verbose parallel pattern:

```
parInit();
for (i=parGetRank(); i <= 42; i+=parGetSize()) {
    parAddTask(“Img:  “+i);         // work segments (tasks) cour
}
parReportTasks();                   // submit created work segments

function work(i) {
    load(i);
    taskNo = floor(i / parGetSize());    // =0,1,2,3,4,5,...
    parReportProgress(taskNo, 20);   // advancement
    ...my (recorded) code…
    parReportProgress(taskNo, 80);
    store(i);
    parReportProgress(taskNo, 100);
}

for (i=parGetRank(); i <= 42; i+=parGetSize()) {
    work(i);
}
parFinalize();
```

TEMPORARY

One may need to use instead

```
function myRank()
{ return parseInt( parGetRank() ); }
function mySize()
{ return parseInt( parGetSize() ); }
```



Task name	Node 0 pr...	Node 1 pr...	Node 2 pr...	Node 3 pr...
Img: 0	✓ Done			
Img: 4	✓ Done			
Img: 8	◌			
Img: 1		✓ Done		
Img: 5		✓ Done		
Img: 9		◌		
Img: 2			✓ Done	
Img: 6			✓ Done	
Img: 3				✓ Done
Img: 7				✓ Done