Recitation 1: Multitasking

Kai Mast

Threads vs. Processes

	Threads	**Processes**
How to start?	`pthread_create()`	`fork()(+exec())`
Own Address Space?	No	Yes
Can share memory?	Yes	No
Can execute concurrently?	Yes	Yes

(and other differences not relevant for the following...)

A primer on concurrency

```
int main() {
    int i = fork();
    if(i == 0)
       printf(“I’m the child! \n”)
    else
       printf(“I’m the parent! \n”);
    return 0;
}
```

Two possible outputs:

```
bash:~ > ./a.out
I’m the child
I’m the parent
```

Or

```
bash:~ > ./a.out
I’m the parent
I’m the child
```

Why?

A primer on concurrency

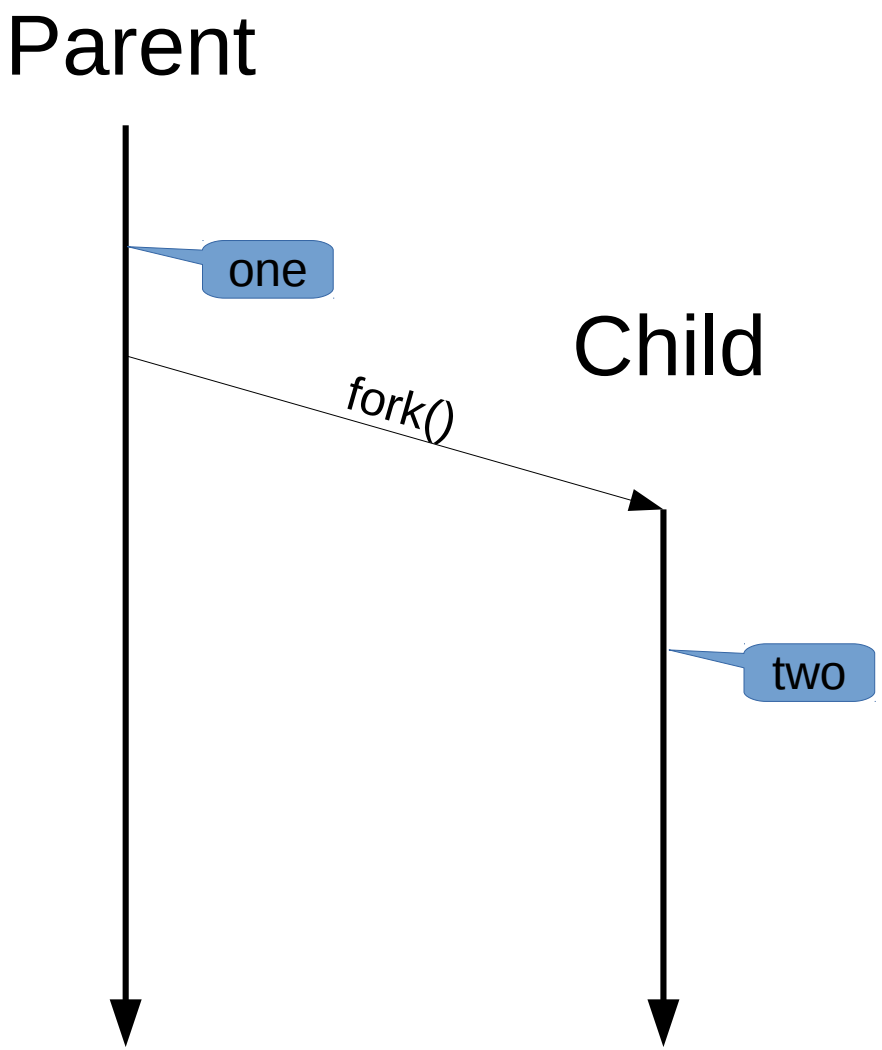
```
int main() {
    printf(“one \n”);
    int i = fork();
    if(i == 0)
       printf(“two \n”)
    return 0;
}
```

Only one possible output:

```
bash:~ > ./a.out
one
two
```

Why?

A visualization of concurrency



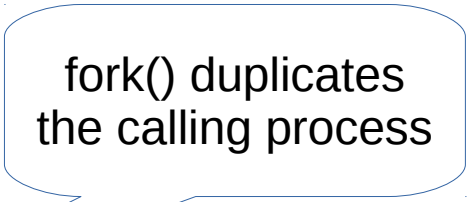
Arrow implies “happens before”

Question 2a)

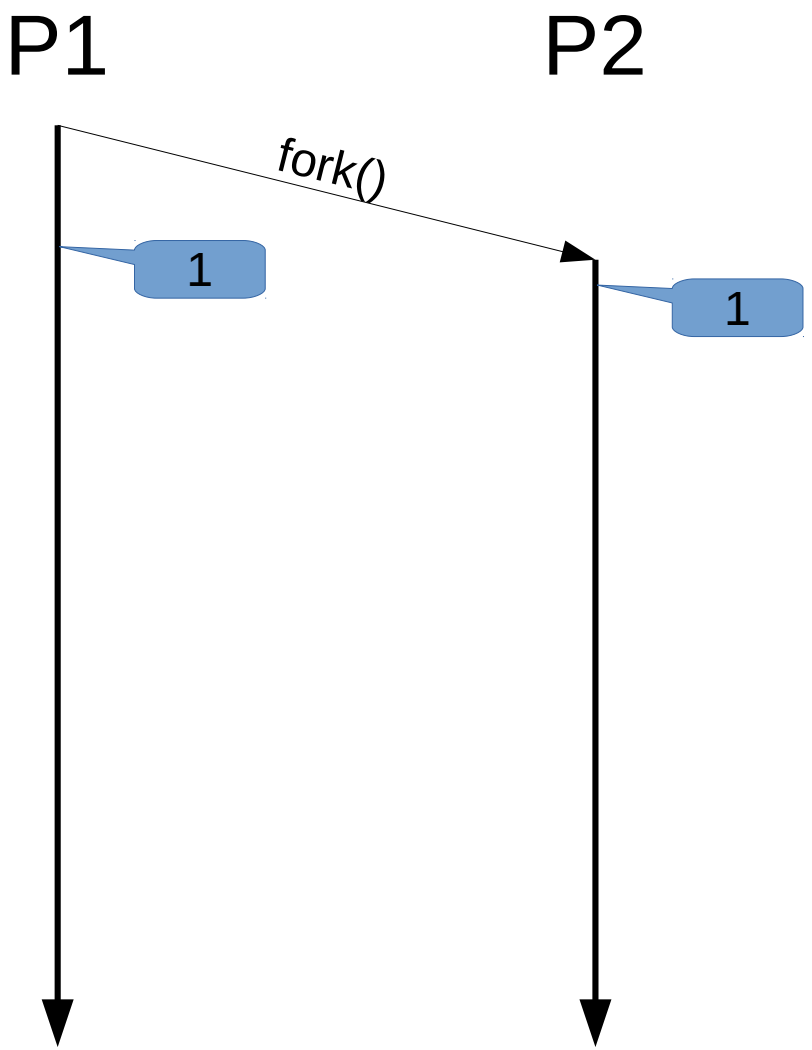
Multiprocessing.

What the Fork?

```
int result = 0;

int main() {
   int i;
   for (i = 0; i < 2; i ++){
      fork();
      result++;
      printf(“result = %d\n”, result);
   }
   printf(“result = %d\n”, result);
   return 0;
}
```

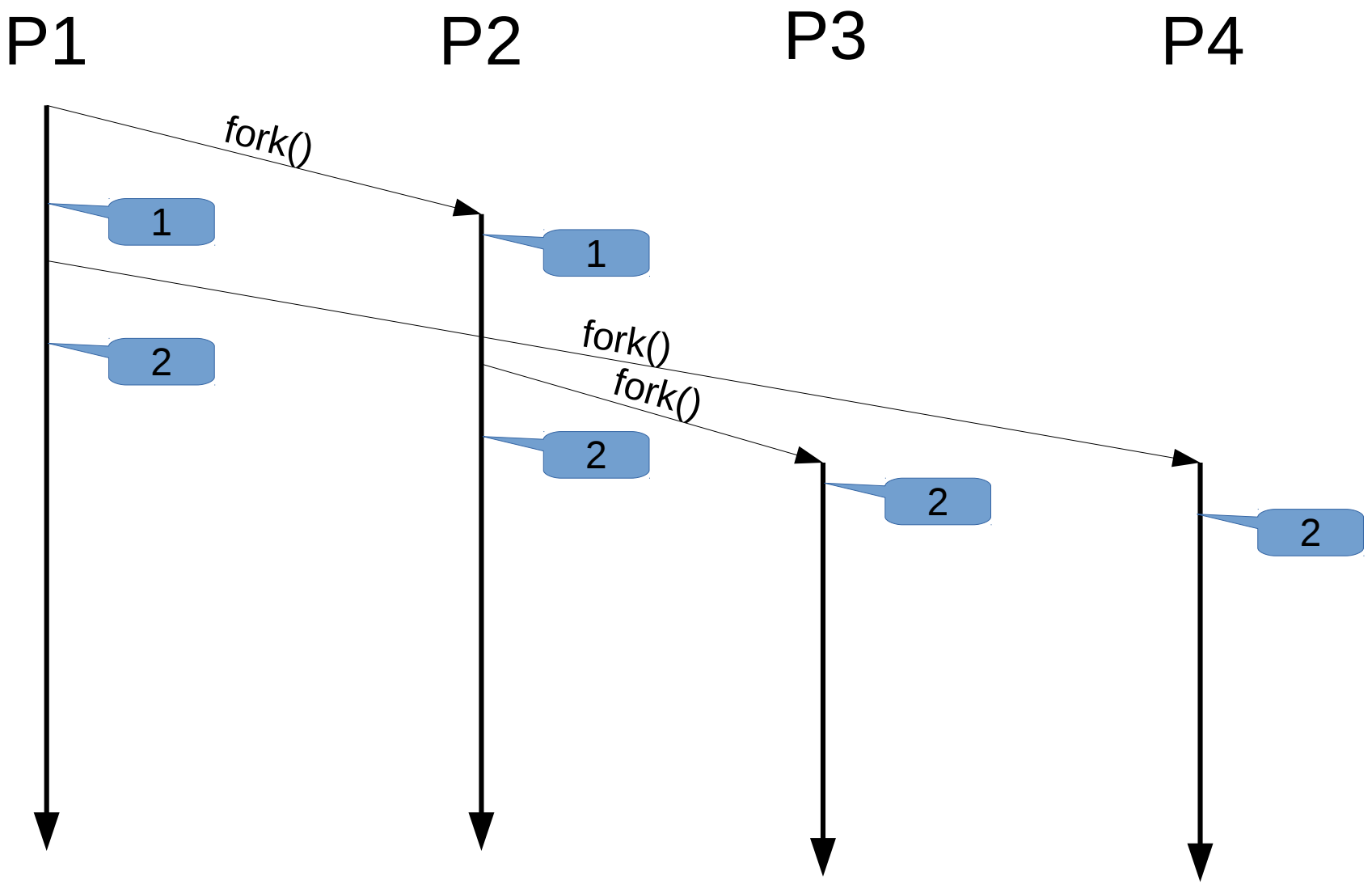


Step 1



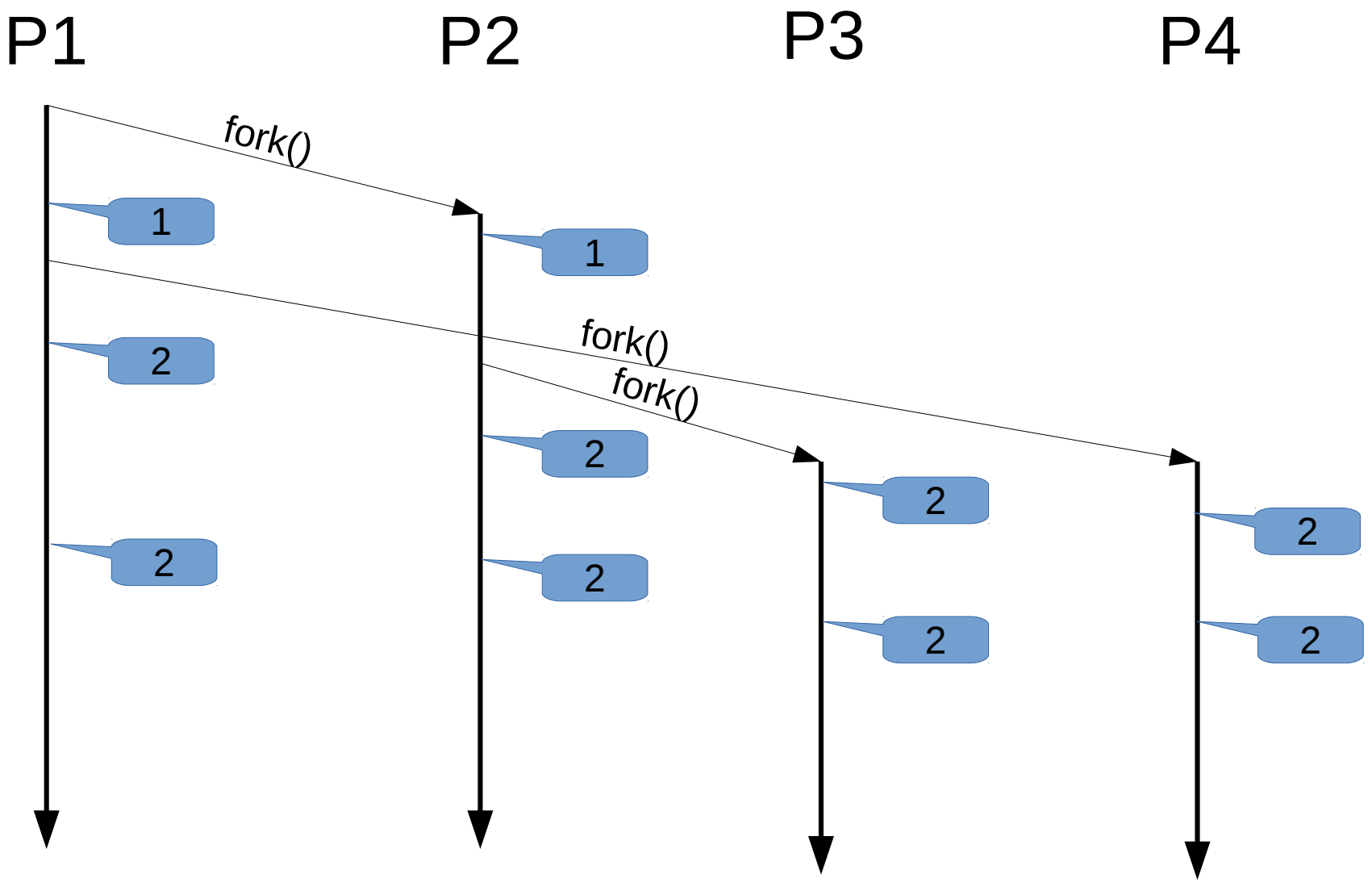
Note: This is only one possible schedule

Step 2



Note: This is only one possible schedule

Step 3



Note: This is only one possible schedule

Question 2b)

Multithreading.

Do not feel threatened by threads

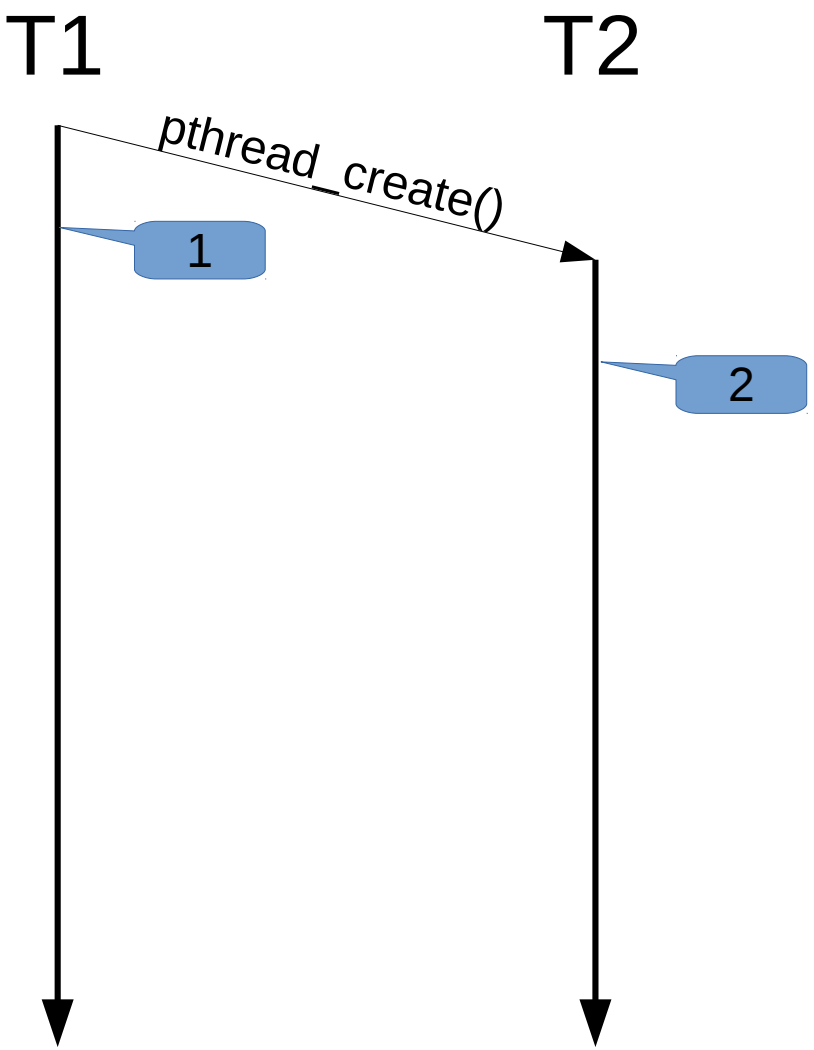
```
int result = 0;
pthread_t tid[2];

void *inc_result(void *ignore) {
   result++;
   printf("result = %d\n", result);
   fflush(stdout);
   return NULL;
}

int main() {
   for (int i = 0; i < 2; i ++){
      pthread create(&tid[i], NULL, &inc_result, NULL);
      result++;
      printf('result = %d\n', result);
   }
   printf("result = %d\n", result);
   return 0;
}
```

result is a global variable

Each child-thread will execute this function

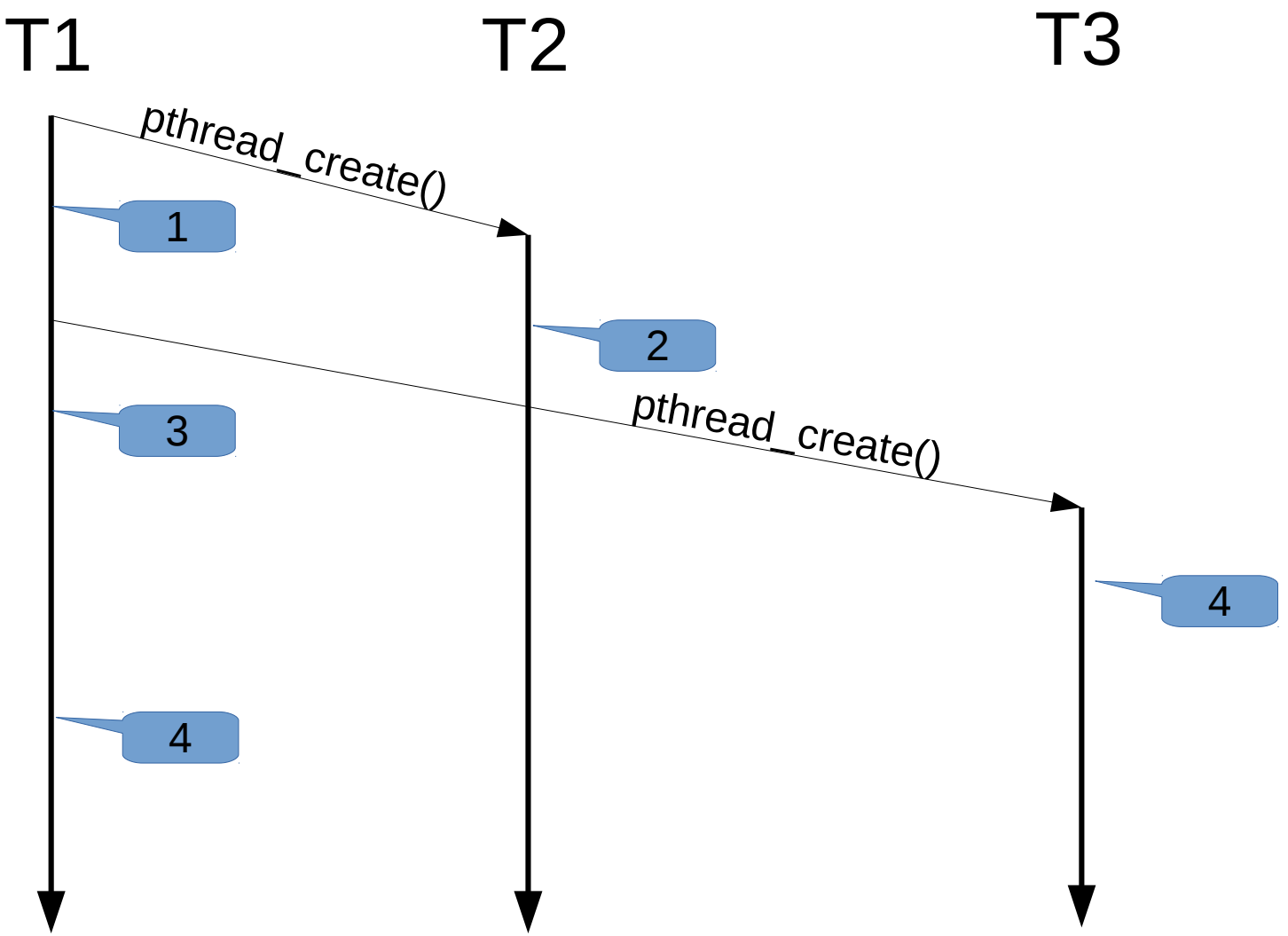
Creates a new thread in the same process

Step 1



Note: This is only one possible schedule

Step 2



Note: This is only one possible schedule