1. The code in vector-deadlock simply adds all the elements of source vector to destination vector. To make the function thread safe, it locks the 2 vectors when adding. But if 2 threads add value from different orders (i.e. t1 adds v0 to v1, t2 adds v1 to v0), deadlock can occur. (Chapter 32, P7)

   For the parameters specified here, it doesn't change from run to run.

2. Not always.
3. -n 1 (A single thread won't cause deadlock)

4. The code avoids deadlock by providing a **total ordering** on lock acquisition, that is always acquiring one lock before another. (Chapter 32, P7) The special case deals with the case where 2 parameters are the same vector, e.g. vector_add(v0, v0).

5.

```
(-add(0, 1)
luyan@LAPTOP-IIIL4GAE:~/chapter32$ ./vector-global-order -t -n 2 -l 100000 -d
Time: 0.03 seconds
luyan@LAPTOP-IIIL4GAE:~/chapter32$ ./vector-global-order -t -n 2 -l 1000000 -d
Time: 0.22 seconds
luyan@LAPTOP-IIIL4GAE:~/chapter32$ ./vector-global-order -t -n 2 -l 10000000 -d
Time: 2.16 seconds
luyan@LAPTOP-IIIL4GAE:~/chapter32$ ./vector-global-order -t -n 2 -l 100000000 -d
Time: 22.44 seconds
luyan@LAPTOP-IIIL4GAE:~/chapter32$ ./vector-global-order -t -n 3 -l 100000 -d
Time: 0.05 seconds
luyan@LAPTOP-IIIL4GAE:~/chapter32$ ./vector-global-order -t -n 4 -l 100000 -d
Time: 0.12 seconds
luyan@LAPTOP-IIIL4GAE:~/chapter32$ ./vector-global-order -t -n 5 -l 100000 -d
Time: 0.14 seconds
luyan@LAPTOP-IIIL4GAE:~/chapter32$ ./vector-global-order -t -n 6 -l 100000 -d
Time: 0.28 seconds
luyan@LAPTOP-IIIL4GAE:~/chapter32$ ./vector-global-order -t -n 7 -l 100000 -d
Time: 0.31 seconds
luyan@LAPTOP-IIIL4GAE:~/chapter32$
```

6.After turning on parallelism (-p), the performance increases by at least 50%.
7. The code uses the **routine pthread mutex trylock()** that either grabs the lock (if it is available) and returns success or returns an error code indicating the lock is held. This method works by checking if a lock (e.g. lock2) that thread 1 wants to acquire is already in use, if yes, thread 1 unlock its held lock (e.g. lock 1) to let lock 2 be unlocked first. However though very unlikely, this approach may suffer from livelock problem. (Chapter 32 P9)

Q: Is the first call to pthread mutex trylock() really needed?
A: The first chunk of trylock is not necessary.

Q:How fast does it run compared to the global order approach?
A: It runs significantly slower than the global order approach.

Q: How does the number of retries, as counted by the code, change as the number of threads increases?

```
luyan@LAPTOP-IIIL4GAE:~/chapter32$ ./vector-try-wait -t -n 2 -l 100000 -d
Retries: 653921
Time: 0.07 seconds
luyan@LAPTOP-IIIL4GAE:~/chapter32$ ./vector-try-wait -t -n 3 -l 100000 -d
Retries: 667300
Time: 0.13 seconds
luyan@LAPTOP-IIIL4GAE:~/chapter32$ ./vector-try-wait -t -n 4 -l 100000 -d
Retries: 2034735
Time: 0.44 seconds
luyan@LAPTOP-IIIL4GAE:~/chapter32$ ./vector-try-wait -t -n 5 -l 100000 -d
Retries: 2902812
Time: 0.70 seconds
luyan@LAPTOP-IIIL4GAE:~/chapter32$ ./vector-try-wait -t -n 6 -l 100000 -d
Retries: 3509625
Time: 1.00 seconds
luyan@LAPTOP-IIIL4GAE:~/chapter32$ ./vector-try-wait -t -n 7 -l 100000 -d
Retries: 4729404
Time: 1.39 seconds
luyan@LAPTOP-IIIL4GAE:~/chapter32$
```

8. This code avoids deadlock by adding a global lock to ensure the atomicity of the lock acquiring and adding.

Performance:

(1) Compared to try-wait without -p: significantly faster

```
luyan@LAPTOP-IIIL4GAE:~/chapter32$ ./vector-try-wait -t -n 2 -l 100000 -d
Retries: 653921
Time: 0.07 seconds
luyan@LAPTOP-IIIL4GAE:~/chapter32$ ./vector-try-wait -t -n 3 -l 100000 -d
Retries: 667300
Time: 0.13 seconds
luyan@LAPTOP-IIIL4GAE:~/chapter32$ ./vector-try-wait -t -n 4 -l 100000 -d
Retries: 2034735
Time: 0.44 seconds
luyan@LAPTOP-IIIL4GAE:~/chapter32$ ./vector-try-wait -t -n 5 -l 100000 -d
Retries: 2902812
Time: 0.70 seconds
luyan@LAPTOP-IIIL4GAE:~/chapter32$ ./vector-try-wait -t -n 6 -l 100000 -d
Retries: 3509625
Time: 1.00 seconds
luyan@LAPTOP-IIIL4GAE:~/chapter32$ ./vector-try-wait -t -n 7 -l 100000 -d
Retries: 4729404
Time: 1.39 seconds
luyan@LAPTOP-IIIL4GAE:~/chapter32$ ./vector-avoid-hold-and-wait -t -n 2 -l 100000 -d
Time: 0.03 seconds
luyan@LAPTOP-IIIL4GAE:~/chapter32$ ./vector-avoid-hold-and-wait -t -n 3 -l 100000 -d
Time: 0.15 seconds
luyan@LAPTOP-IIIL4GAE:~/chapter32$ ./vector-avoid-hold-and-wait -t -n 4 -l 100000 -d
Time: 0.18 seconds
luyan@LAPTOP-IIIL4GAE:~/chapter32$ ./vector-avoid-hold-and-wait -t -n 5 -l 100000 -d
Time: 0.23 seconds
luyan@LAPTOP-IIIL4GAE:~/chapter32$ ./vector-avoid-hold-and-wait -t -n 6 -l 100000 -d
Time: 0.25 seconds
luyan@LAPTOP-IIIL4GAE:~/chapter32$ ./vector-avoid-hold-and-wait -t -n 7 -l 100000 -d
Time: 0.31 seconds
```

(2) Compared to global ordering with -p: slightly slower

```
luyan@LAPTOP-IIIL4GAE:~/chapter32$ ./vector-global-order -t -n 2 -l 100000 -d -p
Time: 0.01 seconds
luyan@LAPTOP-IIIL4GAE:~/chapter32$ ./vector-global-order -t -n 3 -l 100000 -d -p
Time: 0.01 seconds
luyan@LAPTOP-IIIL4GAE:~/chapter32$ ./vector-global-order -t -n 4 -l 100000 -d -p
Time: 0.02 seconds
luyan@LAPTOP-IIIL4GAE:~/chapter32$ ./vector-global-order -t -n 5 -l 100000 -d -p
Time: 0.02 seconds
luyan@LAPTOP-IIIL4GAE:~/chapter32$ ./vector-global-order -t -n 6 -l 100000 -d -p
Time: 0.02 seconds
luyan@LAPTOP-IIIL4GAE:~/chapter32$ ./vector-global-order -t -n 7 -l 100000 -d -p
Time: 0.02 seconds
luyan@LAPTOP-IIIL4GAE:~/chapter32$ ./vector-avoid-hold-and-wait -t -n 2 -l 100000 -d -p
Time: 0.04 seconds
luyan@LAPTOP-IIIL4GAE:~/chapter32$ ./vector-avoid-hold-and-wait -t -n 3 -l 100000 -d -p
Time: 0.05 seconds
luyan@LAPTOP-IIIL4GAE:~/chapter32$ ./vector-avoid-hold-and-wait -t -n 4 -l 100000 -d -p
Time: 0.07 seconds
luyan@LAPTOP-IIIL4GAE:~/chapter32$ ./vector-avoid-hold-and-wait -t -n 5 -l 100000 -d -p
Time: 0.09 seconds
luyan@LAPTOP-IIIL4GAE:~/chapter32$ ./vector-avoid-hold-and-wait -t -n 6 -l 100000 -d -p
Time: 0.12 seconds
luyan@LAPTOP-IIIL4GAE:~/chapter32$ ./vector-avoid-hold-and-wait -t -n 7 -l 100000 -d -p
Time: 0.16 seconds
luyan@LAPTOP-IIIL4GAE:~/chapter32$
```

9. The fetch-and-add (FAA) CPU instruction atomically increments the contents of a memory location by a specified value. Therefore it has similar semantics as using locks.

10.

With -p

```
luyan@LAPTOP-IIIL4GAE:~/chapter32$ ./vector-nolock -t -n 2 -l 100000 -d
Time: 0.54 seconds
luyan@LAPTOP-IIIL4GAE:~/chapter32$ ./vector-nolock -t -n 2 -l 100000 -d -p
Time: 0.09 seconds
luyan@LAPTOP-IIIL4GAE:~/chapter32$ ./vector-nolock -t -n 3 -l 100000 -d -p
Time: 0.10 seconds
luyan@LAPTOP-IIIL4GAE:~/chapter32$ ./vector-nolock -t -n 4 -l 100000 -d -p
Time: 0.06 seconds
luyan@LAPTOP-IIIL4GAE:~/chapter32$ ./vector-nolock -t -n 5 -l 100000 -d -p
Time: 0.09 seconds
luyan@LAPTOP-IIIL4GAE:~/chapter32$ ./vector-nolock -t -n 6 -l 100000 -d -p
Time: 0.11 seconds
luyan@LAPTOP-IIIL4GAE:~/chapter32$ ./vector-nolock -t -n 7 -l 100000 -d -p
Time: 0.11 seconds
```

```
luyan@LAPTOP-IIIL4GAE:~/chapter32$ ./vector-global-order -t -n 2 -l 100000 -d -p
Time: 0.01 seconds
luyan@LAPTOP-IIIL4GAE:~/chapter32$ ./vector-global-order -t -n 3 -l 100000 -d -p
Time: 0.01 seconds
luyan@LAPTOP-IIIL4GAE:~/chapter32$ ./vector-global-order -t -n 4 -l 100000 -d -p
Time: 0.02 seconds
luyan@LAPTOP-IIIL4GAE:~/chapter32$ ./vector-global-order -t -n 5 -l 100000 -d -p
Time: 0.02 seconds
luyan@LAPTOP-IIIL4GAE:~/chapter32$ ./vector-global-order -t -n 6 -l 100000 -d -p
Time: 0.02 seconds
luyan@LAPTOP-IIIL4GAE:~/chapter32$ ./vector-global-order -t -n 7 -l 100000 -d -p
Time: 0.02 seconds
luyan@LAPTOP-IIIL4GAE:~/chapter32$ ./vector-avoid-hold-and-wait -t -n 2 -l 100000 -d -p
Time: 0.04 seconds
luyan@LAPTOP-IIIL4GAE:~/chapter32$ ./vector-avoid-hold-and-wait -t -n 3 -l 100000 -d -p
Time: 0.05 seconds
luyan@LAPTOP-IIIL4GAE:~/chapter32$ ./vector-avoid-hold-and-wait -t -n 4 -l 100000 -d -p
Time: 0.07 seconds
luyan@LAPTOP-IIIL4GAE:~/chapter32$ ./vector-avoid-hold-and-wait -t -n 5 -l 100000 -d -p
Time: 0.09 seconds
luyan@LAPTOP-IIIL4GAE:~/chapter32$ ./vector-avoid-hold-and-wait -t -n 6 -l 100000 -d -p
Time: 0.12 seconds
luyan@LAPTOP-IIIL4GAE:~/chapter32$ ./vector-avoid-hold-and-wait -t -n 7 -l 100000 -d -p
Time: 0.16 seconds
```

Without -p

```
luyan@LAPTOP-IIIL4GAE:~/chapter32$ ./vector-nolock -t -n 2 -l 100000 -d
Time: 0.10 seconds
luyan@LAPTOP-IIIL4GAE:~/chapter32$ ./vector-nolock -t -n 3 -l 100000 -d
Time: 0.43 seconds
luyan@LAPTOP-IIIL4GAE:~/chapter32$ ./vector-nolock -t -n 4 -l 100000 -d
Time: 0.77 seconds
luyan@LAPTOP-IIIL4GAE:~/chapter32$ ./vector-nolock -t -n 5 -l 100000 -d
Time: 0.66 seconds
luyan@LAPTOP-IIIL4GAE:~/chapter32$ ./vector-nolock -t -n 6 -l 100000 -d
Time: 0.77 seconds
luyan@LAPTOP-IIIL4GAE:~/chapter32$ ./vector-nolock -t -n 7 -l 100000 -d
Time: 0.99 seconds
luyan@LAPTOP-IIIL4GAE:~/chapter32$
```

```
luyan@LAPTOP-IIIL4GAE:~/chapter32$ ./vector-global-order -t -n 2 -l 100000 -d
Time: 0.02 seconds
luyan@LAPTOP-IIIL4GAE:~/chapter32$ ./vector-global-order -t -n 3 -l 100000 -d
Time: 0.05 seconds
luyan@LAPTOP-IIIL4GAE:~/chapter32$ ./vector-global-order -t -n 4 -l 100000 -d
Time: 0.12 seconds
luyan@LAPTOP-IIIL4GAE:~/chapter32$ ./vector-global-order -t -n 5 -l 100000 -d
Time: 0.16 seconds
luyan@LAPTOP-IIIL4GAE:~/chapter32$ ./vector-global-order -t -n 6 -l 100000 -d
Time: 0.27 seconds
luyan@LAPTOP-IIIL4GAE:~/chapter32$ ./vector-global-order -t -n 7 -l 100000 -d
Time: 0.31 seconds
```

```
luyan@LAPTOP-IIIL4GAE:~/chapter32$ ./vector-avoid-hold-and-wait -t -n 2 -l 100000 -d
Time: 0.04 seconds
luyan@LAPTOP-IIIL4GAE:~/chapter32$ ./vector-avoid-hold-and-wait -t -n 3 -l 100000 -d
Time: 0.14 seconds
luyan@LAPTOP-IIIL4GAE:~/chapter32$ ./vector-avoid-hold-and-wait -t -n 4 -l 100000 -d
Time: 0.20 seconds
luyan@LAPTOP-IIIL4GAE:~/chapter32$ ./vector-avoid-hold-and-wait -t -n 5 -l 100000 -d
Time: 0.23 seconds
luyan@LAPTOP-IIIL4GAE:~/chapter32$ ./vector-avoid-hold-and-wait -t -n 6 -l 100000 -d
Time: 0.26 seconds
luyan@LAPTOP-IIIL4GAE:~/chapter32$ ./vector-avoid-hold-and-wait -t -n 7 -l 100000 -d
Time: 0.30 seconds
```