





```

tiennk@tiennk-VirtualBox:~$ ./vector-global-order -t -n 2 -l 100000 -d
Time: 0.03 seconds
tiennk@tiennk-VirtualBox:~$ ./vector-global-order -t -n 2 -l 100000 -d
Time: 0.02 seconds
tiennk@tiennk-VirtualBox:~$ ./vector-global-order -t -n 2 -l 100000 -d
Time: 0.03 seconds
tiennk@tiennk-VirtualBox:~$ ./vector-global-order -t -n 2 -l 100000 -d
Time: 0.03 seconds
tiennk@tiennk-VirtualBox:~$ ./vector-global-order -t -n 2 -l 100000 -d
Time: 0.03 seconds
tiennk@tiennk-VirtualBox:~$ ./vector-global-order -t -n 2 -l 100000 -d
Time: 0.02 seconds
tiennk@tiennk-VirtualBox:~$ ./vector-global-order -t -n 2 -l 100000 -d
Time: 0.03 seconds
tiennk@tiennk-VirtualBox:~$ ./vector-global-order -t -n 2 -l 100000 -d
Time: 0.02 seconds
tiennk@tiennk-VirtualBox:~$ ./vector-global-order -t -n 2 -l 100000 -d
Time: 0.03 seconds
tiennk@tiennk-VirtualBox:~$ ./vector-global-order -t -n 2 -l 100000 -d
Time: 0.03 seconds
tiennk@tiennk-VirtualBox:~$ ./vector-global-order -t -n 2 -l 100000 -d
Time: 0.03 seconds

```

Chạy `-t -n 2 -d` với độ dài vòng lặp `1e4`, `1e5`, `1e6` và `1e7` chỉ ra rằng thời gian chạy quy mô gần tuyến tính với độ dài vòng lặp.

```

tiennk@tiennk-VirtualBox:~$ ./vector-global-order -t -n 2 -l 10000 -d
Time: 0.00 seconds
tiennk@tiennk-VirtualBox:~$ ./vector-global-order -t -n 2 -l 100000 -d
Time: 0.02 seconds
tiennk@tiennk-VirtualBox:~$ ./vector-global-order -t -n 2 -l 1000000 -d
Time: 0.19 seconds
tiennk@tiennk-VirtualBox:~$ ./vector-global-order -t -n 2 -l 10000000 -d
Time: 1.65 seconds
tiennk@tiennk-VirtualBox:~$ ./vector-global-order -t -n 2 -l 10000000 -d

```

Việc chạy `-t -l 100000 -d` với số threads 2, 20 và 99 cho biết rằng thời gian chạy sẽ thay đổi tương đối tuyến tính với số thread.

```

tiennk@tiennk-VirtualBox:~$ ./vector-global-order -t -n 2 -l 100000 -d
Time: 0.03 seconds
tiennk@tiennk-VirtualBox:~$ ./vector-global-order -t -n 20 -l 100000 -d
Time: 0.19 seconds
tiennk@tiennk-VirtualBox:~$ ./vector-global-order -t -n 99 -l 100000 -d
Time: 0.85 seconds

```

6. Điều gì xảy ra nếu bạn bật cờ song song (`-p`)? Bạn mong đợi hiệu suất sẽ thay đổi bao nhiêu khi mỗi thread đang làm việc để thêm các vector khác nhau (đó là những gì `-p` cho phép) so với làm việc trên những vector giống nhau?

Tôi hy vọng rằng chương trình sẽ hoàn thành nhanh chóng hơn, nhưng tôi không biết nhanh hơn bao nhiêu. Tôi hy vọng rằng khi bật tính năng song song, thời gian chạy sẽ tiếp tục mở rộng tuyến tính với độ dài vòng lặp. Thú vị hơn, tôi mong đợi nó sẽ bất biến đối với số thread, lên đến số bộ xử lý logic và sau đó sẽ chia tỷ lệ tuyến tính (theo bội số của số bộ xử lý logic) quá thời điểm đó.

Với hai luồng ( -n 2), tốc độ tăng khoảng 7-8 lần khi bật chế độ song song ( -p) và con số này vẫn phù hợp với các độ dài vòng lặp khác nhau. Thay đổi số luồng thay đổi thời gian chạy như sau:

2 chủ đề: ~ 0.03 giây

3 chủ đề: ~ 0.04 giây

4 chủ đề: ~ 0.05 giây

5 chủ đề: ~ 0.07 giây

6 chủ đề: ~ 0.08 giây

7 chủ đề: ~ 0.10 giây

8 chủ đề: ~ 0.09 giây

16 chủ đề: ~ 0.16 giây

32 chủ đề: ~ 0.28 giây

64 chủ đề: ~ 0.57 giây

```
tiennk@tiennk-VirtualBox:~$ ./vector-global-order -t -p 2 -l 100000 -d
Time: 0.03 seconds
tiennk@tiennk-VirtualBox:~$ ./vector-global-order -t -n 2 -l 100000 -d -p
Time: 0.04 seconds
tiennk@tiennk-VirtualBox:~$ ./vector-global-order -t -n 3 -l 100000 -d -p
Time: 0.04 seconds
tiennk@tiennk-VirtualBox:~$ ./vector-global-order -t -n 4 -l 100000 -d -p
Time: 0.05 seconds
tiennk@tiennk-VirtualBox:~$ ./vector-global-order -t -n 5 -l 100000 -d -p
Time: 0.07 seconds
tiennk@tiennk-VirtualBox:~$ ./vector-global-order -t -n 6 -l 100000 -d -p
Time: 0.08 seconds
tiennk@tiennk-VirtualBox:~$ ./vector-global-order -t -n 7 -l 100000 -d -p
Time: 0.10 seconds
tiennk@tiennk-VirtualBox:~$ ./vector-global-order -t -n 8 -l 100000 -d -p
Time: 0.09 seconds
tiennk@tiennk-VirtualBox:~$ ./vector-global-order -t -n 16 -l 100000 -d -p
Time: 0.16 seconds
tiennk@tiennk-VirtualBox:~$ ./vector-global-order -t -n 32 -l 100000 -d -p
Time: 0.28 seconds
tiennk@tiennk-VirtualBox:~$ ./vector-global-order -t -n 64 -l 100000 -d -p
Time: 0.57 seconds
```



Những con số này không khớp chính xác với dự đoán của tôi, nhưng chúng không quá xa. Chia tỷ lệ tuyến tính dường như không có hiệu lực cho đến khi chúng tôi đạt được 8 chủ đề. Đối với 5 luồng trở xuống, thời gian chạy gần bằng hằng số và tỷ lệ là tuyến tính phụ đối với ít hơn 8 luồng.

**7. Bây giờ chúng ta hãy nghiên cứu vector-try-wait.c. Trước tiên, hãy đảm bảo rằng bạn hiểu mã. Cuộc gọi đầu tiên có pthread\_mutex\_trylock() thực sự cần thiết không? Bây giờ hãy chạy mã. Nó chạy nhanh như thế nào so với cách tiếp cận đặt hàng toàn cầu? Làm thế nào để số lần thử lại, được tính bằng mã, thay đổi khi số lượng chủ đề tăng lên?**

Cuộc gọi đầu tiên thực sự cần thiết, bạn còn định khóa vector đích như thế nào nữa?

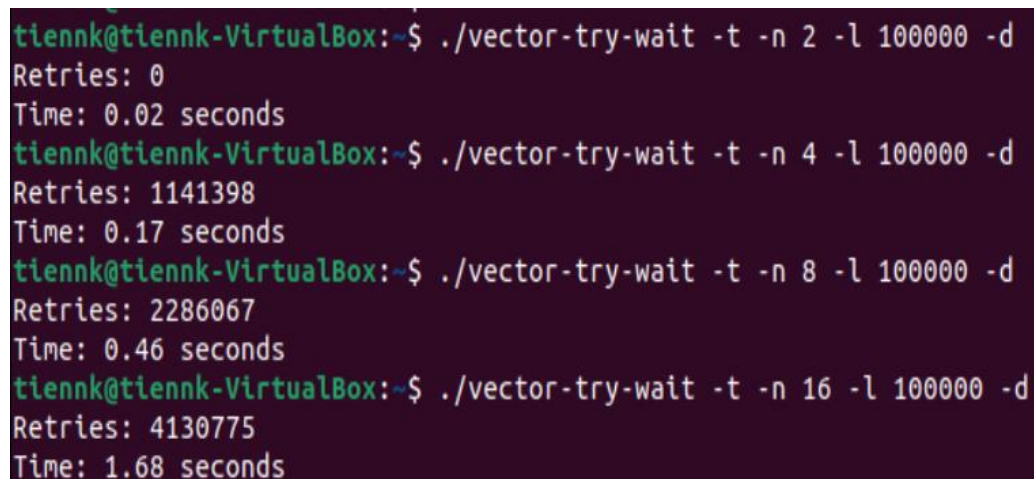
Dưới đây là thời gian chạy với các tùy chọn -t -l 100000 -d, thay đổi số lượng luồng ( -n):

2 chủ đề: ~ 0,02 giây, ~ 0 lần thử lại

4 chủ đề: ~ 0,17 giây, ~ 1141398 lần thử lại

8 luồng: ~0.46 giây, ~ 2286067 lần thử lại

16 luồng: ~ 1.68 giây, ~ 4130775 lần thử lại



```
tiennk@tiennk-VirtualBox:~$ ./vector-try-wait -t -n 2 -l 100000 -d
Retries: 0
Time: 0.02 seconds
tiennk@tiennk-VirtualBox:~$ ./vector-try-wait -t -n 4 -l 100000 -d
Retries: 1141398
Time: 0.17 seconds
tiennk@tiennk-VirtualBox:~$ ./vector-try-wait -t -n 8 -l 100000 -d
Retries: 2286067
Time: 0.46 seconds
tiennk@tiennk-VirtualBox:~$ ./vector-try-wait -t -n 16 -l 100000 -d
Retries: 4130775
Time: 1.68 seconds
```

Dưới đây là thời gian chạy dưới độ song song cao -t -l 100000 -d -p(theo cấu trúc, tất cả các lần chạy đều không có lần thử lại):

2 chủ đề: ~ 0,03 giây

4 chủ đề: ~ 0,06 giây

8 chủ đề: ~ 0,09 giây

16 chủ đề: ~0.15 giây

32 chủ đề: ~ 0,29 giây

64 chủ đề: ~ 0,55 giây

```

tiennk@tiennk-VirtualBox:~$ ./vector-try-wait -t -n 2 -l 100000 -d -p
Retries: 0
Time: 0.03 seconds
tiennk@tiennk-VirtualBox:~$ ./vector-try-wait -t -n 4 -l 100000 -d -p
Retries: 0
Time: 0.06 seconds
tiennk@tiennk-VirtualBox:~$ ./vector-try-wait -t -n 8 -l 100000 -d -p
Retries: 0
Time: 0.09 seconds
tiennk@tiennk-VirtualBox:~$ ./vector-try-wait -t -n 16 -l 100000 -d -p
Retries: 0
Time: 0.15 seconds
tiennk@tiennk-VirtualBox:~$ ./vector-try-wait -t -n 32 -l 100000 -d -p
Retries: 0
Time: 0.29 seconds
tiennk@tiennk-VirtualBox:~$ ./vector-try-wait -t -n 64 -l 100000 -d -p
Retries: 0
Time: 0.55 seconds

```

Dưới sự tranh cãi cao, cả thời gian thực hiện và số lần thử lại đều có quy mô siêu tuyến tính (có vẻ như quy mô thời gian nhanh hơn bậc hai, nhưng chậm hơn theo cấp số nhân với cơ số 2). Mặc dù trong ký hiệu O lớn, tỷ lệ này kém hơn nhiều so với giải pháp thứ tự khóa toàn cục, nhưng giải pháp này nhanh hơn đáng kể trong thời gian tuyệt đối đối với số lượng chủ đề nhỏ. Nó nhanh hơn ít nhất 5 lần cho 2 luồng và giải pháp thứ tự khóa toàn cầu chỉ bắt kịp khoảng 11 luồng.

Dưới chế độ song song cao (với -p) thời gian chạy là hiện tượng, và vượt xa vector-global-order lần đầu tiên, và sau đó đối với số lượng chủ đề cao hơn, hai bậc của cường độ.

**8. Bây giờ chúng ta hãy nhìn vào vector-avoid-hold-and-wait.c. Vấn đề chính của cách tiếp cận này là gì? Hiệu suất của nó như thế nào so với các phiên bản khác, khi chạy cả khi có -pvà không có nó?**

Vấn đề chính của cách tiếp cận này là nó quá thô: khóa toàn cục (bảo vệ việc mua lại tất cả các khóa khác) sẽ bị tranh chấp ngay cả khi các vectơ được thao tác bởi mỗi luồng là khác nhau.

-t -l 100000 -d, số lượng chủ đề khác nhau -n:

2 chủ đề: ~ 0,02 giây

4 chủ đề: ~ 0,06 giây

8 chủ đề: ~ 0,11 giây

16 luồng: ~0,18 giây

32 chủ đề: ~ 0,33 giây

```

tiennk@tiennk-VirtualBox:~$ ./vector-avoid-hold-and-wait -t -n 2 -l 100000 -d
Time: 0.02 seconds
tiennk@tiennk-VirtualBox:~$ ./vector-avoid-hold-and-wait -t -n 4 -l 100000 -d
Time: 0.06 seconds
tiennk@tiennk-VirtualBox:~$ ./vector-avoid-hold-and-wait -t -n 8 -l 100000 -d
Time: 0.11 seconds
tiennk@tiennk-VirtualBox:~$ ./vector-avoid-hold-and-wait -t -n 16 -l 100000 -d
Time: 0.18 seconds
tiennk@tiennk-VirtualBox:~$ ./vector-avoid-hold-and-wait -t -n 32 -l 100000 -d
Time: 0.33 seconds

```

-t -l 100000 -d -p (với song song), số lượng chủ đề khác nhau -n:

2 chủ đề: ~ 0,04 giây

4 chủ đề: ~ 0,06 giây

8 chủ đề: ~ 0,10 giây

16 luồng: ~ 0,19 giây

32 chủ đề: ~ 0,32 giây

```

tiennk@tiennk-VirtualBox:~$ ./vector-avoid-hold-and-wait -t -n 2 -l 100000 -d -p
Time: 0.04 seconds
tiennk@tiennk-VirtualBox:~$ ./vector-avoid-hold-and-wait -t -n 4 -l 100000 -d -p
Time: 0.06 seconds
tiennk@tiennk-VirtualBox:~$ ./vector-avoid-hold-and-wait -t -n 8 -l 100000 -d -p
Time: 0.10 seconds
tiennk@tiennk-VirtualBox:~$ ./vector-avoid-hold-and-wait -t -n 16 -l 100000 -d -p
Time: 0.19 seconds
tiennk@tiennk-VirtualBox:~$ ./vector-avoid-hold-and-wait -t -n 32 -l 100000 -d -p
Time: 0.32 seconds

```

Thời gian chạy dưới sự tranh chấp nặng nề (không có -p) rất gần với những người cho vector-global-order. Đối với trường hợp độ song song cao (với -p), vector-avoid-hold-and-wait thực hiện nhiều hơn hoặc ít hơn so với trường hợp độ song song cao, nhưng chậm hơn khoảng hai lần so với trường hợp vector-global-order độ song song cao. Do đó, sự so sánh giữa vector-global-order và vector-try-wait nhiều hơn hoặc ít hơn cũng có ở đây.

**9. Cuối cùng, chúng ta hãy nhìn vào vector-nolock.c. Phiên bản này hoàn toàn không sử dụng khóa; nó có cung cấp ngữ nghĩa chính xác giống như các phiên bản khác không? Tại sao hoặc tại sao không?**

Không. Nó chỉ là nguyên tử đối với việc thêm một cặp mục nhập, không phải mọi cặp mục nhập.