```cpp
template<typename T>
class mc_concurrentqueue
{
public:
    mc_concurrentqueue (size_t capacity)
        : queue (capacity)
    {
    }

    bool try_push (const T& v)
    {
        return queue.try_enqueue (v);
    }

    bool try_pop (T& v)
    {
        return queue.try_dequeue (v);
    }

private:
    moodycamel::ConcurrentQueue<T> queue;
};
```

```
template<typename T>
class mc_concurrentqueue_sized
{
public:
    mc_concurrentqueue_sized (size_t capacity)
        : queue (capacity)
    {
    }

    bool try_push (const T& v)
    {
        if (! queue.try_enqueue (v))
            return false;

        size.fetch_add (1, std::memory_order_release);
        return true;
    }

    bool try_pop (T& v)
    {
        if (! queue.try_dequeue (v))
            return false;

        size.fetch_sub (1, std::memory_order_release);
        return true;
    }

private:
    moodycamel::ConcurrentQueue<T> queue;
    std::atomic<size_t> size { 0 };
};
```

```cpp
template<typename T>
class mc_concurrentqueue
{
public:
    mc_concurrentqueue (size_t capacity)
        : queue (capacity)
    {
    }

    bool try_push (const T& v)
    {
        return queue.try_enqueue (v);
    }

    bool try_pop (T& v)
    {
        return queue.try_dequeue (v);
    }

private:
    moodycamel::ConcurrentQueue<T> queue;
};
```

```cpp
template<typename T>
class mc_concurrentqueue_sized
{
public:
    mc_concurrentqueue_sized (size_t capacity)
        : queue (capacity)
    {
    }

    bool try_push (const T& v)
    {
        if (! queue.try_enqueue (v))
            return false;

        size.fetch_add (1, std::memory_order_release);
        return true;
    }

    bool try_pop (T& v)
    {
        if (! queue.try_dequeue (v))
            return false;

        size.fetch_sub (1, std::memory_order_release);
        return true;
    }

private:
    moodycamel::ConcurrentQueue<T> queue;
    std::atomic<size_t> size { 0 };
};
```

8.00E+06
6.00E+06
4.00E+06
2.00E+06
0.00E+00
drow_mpmc_v6
tbb_concurrent_queue
mc_concurrentqueue
mc_concurrentqueue_sized
farbot_mpmc
farbot_mpmc_wf
rigtorp_mpmc
es_mpmc_queue
choc_mpmc