Текст программы (параллельной):

```
#include <iostream>
#include <queue>
#include <tuple>
#include <condition_variable>
#include <thread>
#include <string>
#include <algorithm>
#include <ostream>
#include <fstream>



using namespace std;
using namespace chrono_literals;



queue<string> q;
mutex mut;
condition_variable cv;
bool finished{ false };

ifstream in("in.txt");
ifstream in_seq("in_seq.txt");
ofstream out("out.txt");
ofstream out_seq("out_seq.txt");



void producer()
{
   string line;
   while (in >> line)
   {
      {
         lock_guard<mutex> lk{ mut };
         q.push(line);
      }

      cv.notify_all();
   }

   {
      lock_guard<mutex> lk{ mut };
      finished = true;
   }



   cv.notify_all();
}



static void consumer()
{
   std::string line;

   while (!finished)
   {
      unique_lock<mutex> l{ mut };
      cv.wait(l, [] { return !q.empty() || finished; });
      while (!q.empty())
```

```
        {
            line = q.front();
            std::reverse(line.begin(), line.end());
            out << line << '\n';

            q.pop();
        }
    }
}



int main()
{
    auto start = std::chrono::high_resolution_clock::now();

    thread t1{ producer };
    thread t2{ consumer };
    t1.join();
    t2.join();

    auto stop = std::chrono::high_resolution_clock::now();
    auto duration = std::chrono::duration_cast<std::chrono::milliseconds>(stop - start);

    std::cout << "Time " << duration.count()
            << " ms" << std::endl;

    std::cout << "finished!\n";
```

Последовательная программа:

```
    auto start_seq = std::chrono::high_resolution_clock::now();
    std::string s;
    while (in_seq >> s)
    {
        std::reverse(s.begin(), s.end());
        out_seq << s << '\n';
    }

    auto stop_seq = std::chrono::high_resolution_clock::now();
    auto duration_seq = std::chrono::duration_cast<std::chrono::milliseconds>(stop_seq - start_seq);

    std::cout << "Time " << duration_seq.count()
            << " ms seq" << std::endl;

    in.close();
    out.close();

    return 0;
}
```

Результаты экспериментов:

Размерность задачи	Ускорение	Последовательная программа, мс	Параллельная программа, мс
6 * 10^6	0.256989	1890,02466	6589,25656

10^8	0.2856565	2305,21354	7950,24565
2,3 * 10^8	0.2694465	2195,23365	8235,65512