

Pthread Report from team20

Fifth assignment of course Operating System

Professor: 周志遠

Part 1: member list and contributions

Member list

- 109062274 資訊工程系三年級 楊子慶 Eroiko
- 109080076 生科院學士班三年級 俞政佑 Blue

Contributions

Items	楊子慶	俞政佑
Basic & Experiment Implementation	V	
Implementation report	V	
Experiment report		V

Explain the implementation Part

TSQueue implementation

`TS_Queue::TS_Queue`

`TS_Queue` 物件要管理幾個 counter, buffer, lock 和 condition variables。

故我們要初始化 counter 和 buffer:

```
1  buffer = new T[buffer_size];
2  size = 0;
3  head = buffer_size - 1;
4  tail = 0;
```

初始化 lock, 使用 `pthread_mutex_init` 函式:

```
1  // initialize mutex for critical section
2  pthread_mutex_init(&mutex, NULL);
```

初始化 condition variables, 使用 `pthread_cond_init` 函式:

```
1  // initialize conditional variables
2  pthread_cond_init(&cond_enqueue, nullptr);
3  pthread_cond_init(&cond_dequeue, nullptr);
```

故總體為:

```

1  TSQueue<T>::TSQueue(int buffer_size) : buffer_size(buffer_size) {
2      // TODO: implements TSQueue constructor
3      // initialize members
4      buffer = new T[buffer_size];
5      size = 0;
6      head = buffer_size - 1;
7      tail = 0;
8      // initialize mutex for critical section
9      pthread_mutex_init(&mutex, NULL);
10     // initialize conditional variables
11     pthread_cond_init(&cond_enqueue, nullptr);
12     pthread_cond_init(&cond_dequeue, nullptr);
13 }

```

TS_Queue::~~TS_Queue

與建構子對應, 我們要刪除必要歸還的成員:

```

1  // free members
2  delete [] buffer;

```

同時要歸還 lock 和 condition variables:

```

1  // destroy condition variables
2  pthread_cond_destroy(&cond_enqueue);
3  pthread_cond_destroy(&cond_dequeue);
4  // delete the mutex
5  pthread_mutex_destroy(&mutex);

```

故總體為:

```

1  template <class T>
2  TSQueue<T>::~~TSQueue() {
3      // TODO: implements TSQueue destructor
4      // free members
5      delete [] buffer;
6      // destroy condition variables
7      pthread_cond_destroy(&cond_enqueue);
8      pthread_cond_destroy(&cond_dequeue);
9      // delete the mutex
10     pthread_mutex_destroy(&mutex);
11 }

```

TS_Queue::enqueue

插入元素會更改 queue, 故我們應該在在新增資料時鎖定本物件, 進入 critical section。

```

1  pthread_mutex_lock(&mutex); // enter critical section
2  // ***** critical section *****
3  //...
4  //...
5  //...
6  // ***** critical section *****
7  pthread_mutex_unlock(&mutex); // leave critical section

```

在 critical section 中, 只有在本資結還塞的下東西時才可以新增資料, 故規定 `cond_enqueue` 這個 condition variable 負責鎖定 `TSQueue::enqueue` 操作, 若 queue 已經額滿, 則 wait 直到解除鎖定。解除鎖定的條件是: 本資結還放得下東西, 故當隊列已經滿了, 使用回圈等待通知說有空位:

```

1  // ***** critical section *****
2  // ...
3  // block if the queue is full
4  while (size == buffer_size)
5      pthread_cond_wait(&cond_enqueue, &mutex);
6  // ...
7  // ***** critical section *****

```

若有空位,則可以放入新資料,並正確的調整 counter:

```
1 // ***** critical section *****
2 // ...
3 // enqueue, size < buffer_size
4 buffer[tail] = item;
5 tail = (tail + 1) % buffer_size;
6 ++size;
7 // ...
8 // ***** critical section *****
```

我們加入新元素後,隊列即不為空當隊列有資料時, `TSQueue::dequeue` 始得完成其行為,故我們應該通知懸停在等待 `cond_dequeue` 條件的執行緒可以繼續進行他們的工作:

```
1 // ***** critical section *****
2 // ...
3 // notify dequeue since we have at least one element
4 pthread_cond_signal(&cond_dequeue);
5 // ...
6 // ***** critical section *****
```

而後離開 critical section。所以整體程式碼可以實作為:

```
1 template <class T>
2 void TSQueue<T>::enqueue(T item) {
3     // TODO: enqueues an element to the end of the queue
4     pthread_mutex_lock(&mutex); // enter critical section
5     // ***** critical section *****
6     // block if the queue is full
7     while (size == buffer_size)
8         pthread_cond_wait(&cond_enqueue, &mutex);
9     // enqueue, size < buffer_size
10    buffer[tail] = item;
11    tail = (tail + 1) % buffer_size;
12    ++size;
13    // notify dequeue since we have at least one element
14    pthread_cond_signal(&cond_dequeue);
```

```

15     // ***** critical section *****
16     pthread_mutex_unlock(&mutex); // leave critical section
17 }

```

TS_Queue::dequeue

起除元素會更改 queue, 故我們應該在在取出資料時鎖定本物件, 進入 critical section。

```

1  pthread_mutex_lock(&mutex); // enter critical section
2  // ***** critical section *****
3  // ...
4  // ...
5  // ...
6  // ***** critical section *****
7  pthread_mutex_unlock(&mutex); // leave critical section

```

在 critical section 中, 只有在本資結有東西可以取出時才可以取出資料, 故規定 `cond_dequeue` 這個 condition variable 負責鎖定 `TSQueue::dequeue` 操作, 若 queue 為空, 則 wait 直到解除鎖定。解除鎖定的條件是: 本資結至少有一個東西可以取出, 故當隊列為空, 使用回圈等待通知說有可取出的物件:

```

1  // ***** critical section *****
2  // ...
3  // block if no element to be dequeue
4  while (!size)
5      pthread_cond_wait(&cond_dequeue, &mutex);
6  // ...
7  // ***** critical section *****

```

若有資料, 則取出資料, 並正確的調整 counter:

```

1 // ***** critical section *****
2 // ...
3 // dequeue, size >= 1
4 head = (head + 1) % buffer_size;
5 ret = buffer[head];
6 --size;
7 // ...
8 // ***** critical section *****

```

我們取出一個元素後, 隊列至少有一個空位, `TSQueue::enqueue` 始得完成其行為, 故我們應該通知懸停在等待 `cond_enqueue` 條件的執行緒可以繼續進行他們的工作:

```

1 // ***** critical section *****
2 // ...
3 // notify enqueue since we have a least one empty space
4 pthread_cond_signal(&cond_enqueue);
5 // ...
6 // ***** critical section *****

```

而後離開 critical section。所以整體程式碼可以實作為:

```

1 template <class T>
2 T TSQueue<T>::dequeue() {
3     // TODO: dequeues the first element of the queue
4     T ret; // element to be dequeued
5     pthread_mutex_lock(&mutex); // enter critical section
6     // ***** critical section *****
7     // block if no element to be dequeue
8     while (!size)
9         pthread_cond_wait(&cond_dequeue, &mutex);
10    // dequeue, size >= 1
11    head = (head + 1) % buffer_size;
12    ret = buffer[head];
13    --size;
14    // notify enqueue since we have a least one empty space
15    pthread_cond_signal(&cond_enqueue);
16    // ***** critical section *****

```

```
17     pthread_mutex_unlock(&mutex); // leave critical section
18     return ret;
19 }
```

TS_Queue::get_size

本方法不更動物件本身, 無需進入 critical section, 直接回傳 `TSQueue::size` 的值:

```
1  template <class T>
2  int TSQueue<T>::get_size() {
3      // TODO: returns the size of the queue
4      return size;
5  }
```

Writer implementation

Writer::start

使用 `pthread_create` 建立執行緒, 並指定執行緒要做的 routine 為 `static Writer::process`。

```
1  void Writer::start() {
2      // TODO: starts a Writer thread
3      pthread_create(&t, nullptr, Writer::process, (void *) this);
4  }
```

Writer::process

仿造 `Reader::process` 靜態方法, 若還沒寫入足夠數量的資料, 就遞減紀錄寫入數量 counter, 並呼叫 `output_queue` 的 `TSQueue::dequeue` 方法取出資料寫入, 結束後回傳空指標。


```

1 void* Writer::process(void* arg) {
2     // TODO: implements the Writer's work
3     auto writer = (Writer *) arg;
4     while (writer->expected_lines--)
5         writer->ofs << *writer->output_queue->dequeue();
6     return nullptr;
7 }

```

Producer implementation

Producer::start

使用 `pthread_create` 建立執行緒, 並指定執行緒要做的 routine 為 `static Producer::process` 。

```

1 void Producer::start() {
2     // TODO: starts a Producer thread
3     pthread_create(&t, nullptr, Producer::process, (void *) this);
4 }

```

Producer::process

`Producer` 的任務是無限的將 `Producer::input_queue` 的資料透過 `Transformer::producer_transform` 來生產資料並放入 `Producer::worker_queue`, 另外要注意 Heap 記憶體管理的議題: 刪除不再使用的物件。

```

1 void* Producer::process(void* arg) {
2     // TODO: implements the Producer's work
3     auto producer = (Producer *) arg;
4     while (true) {
5         if (producer->input_queue->get_size()) {
6             Item * it = producer->input_queue->dequeue(); // item
// to be processed and deleted
7             auto val = producer->transformer-
>producer_transform(it->opcode, it->val); // new value
8             producer->worker_queue->enqueue(new Item(it->key, val,
it->opcode));
9             delete it;
10        }
11    }
12    return nullptr;
13 }

```

Consumer implementation

Consumer::start

使用 `pthread_create` 建立執行緒, 並指定執行緒要做的 routine 為 `static Consumer::process`。

```

1 void Consumer::start() {
2     // TODO: starts a Consumer thread
3     pthread_create(&t, nullptr, Consumer::process, (void *) this);
4 }

```

Consumer::cancel

將旗標 `is_cancel` 設為零, 通知 `static Consumer::process` 結束其無限迴圈並刪除 `Consumer` 實例。

```

1  int Consumer::cancel() {
2      // TODO: cancels the consumer thread
3      is_cancel = true;
4      return pthread_cancel(t);
5  }

```

Consumer::process

我們要實作的部分與 `static Producer::process` 幾乎相同, 只是改由 `wConsumer::worker_queue` 取出, 由 `Transformer::consumer_transform` 轉換資料後放入 `Consumer::output_queue` 中。

```

1  // inside Consumer::process static method
2  while (!consumer->is_cancel) {
3      pthread_setcancelstate(PTHREAD_CANCEL_DISABLE, nullptr);
4
5      // TODO: implements the Consumer's work
6      if (consumer->worker_queue->get_size()) {
7          Item * it = consumer->worker_queue->dequeue();
8          auto val = consumer->transformer->consumer_transform(it-
9              >opcode, it->val);
10             consumer->output_queue->enqueue(new Item(it->key, val, it-
11                 >opcode));
12             delete it;
13         }
14
15     pthread_setcancelstate(PTHREAD_CANCEL_ENABLE, nullptr);
16 }

```

ConsumerController implementation

為了較為精確地實現計時器的功能, 我採用 `sigaction` 和 `timer` 的 API, 這使我要另外 include `signal.h` 和 `sys/time.h` 這兩個標頭檔, 並為 `ConsumerController` 物件新增一個靜態成員 `global_ctr` 和靜態方法 `static ConsumerController::handler`。

另外, 為了方便紀錄實驗數據, 我定義了兩個不影響整體邏輯的全域變數: `rec_ptr`, `print_scaling_msg` 讓我們可以更方便地做實驗, 這個細節將留到實驗講解來說明。

ConsumerController::~~ConsumerController

原本 `TODO` 並沒有這一塊的修改, 不過為了實現計時器的功能, 我需要在 `ConsumerController` 結束其生命週期時將靜態成員 `global_ctr` 重設為 `nullptr` 以確保 `ConsumerController::process` 可以如預期般結束無窮迴圈。

```
1 ConsumerController::~~ConsumerController() {  
2     if (global_ctr == this)  
3         global_ctr = nullptr;  
4 }
```

ConsumerController::start

為了紀錄實驗數據, 首先要先初始化 `rec_ptr` 這個紀錄 `ConsumerController` scaling 情況的 `vector<int> *`。

當外界選擇為此全域向量指標分配記憶體空間 (即不為 `nullptr`), 表示我們要紀錄實驗數據, 故在本方法中重設此向量指標, 並加入當前的 Consumer 數量 (一開始必為 `0`)。

接著要初始化我們的靜態成員 `ConsumerController::global_ctr`, 將其設為實例自身, 以便後續計時器可以順利運作。

最後使用 `pthread_create` 建立執行緒, 並指定執行緒要做的 routine 為 `static ConsumerController::process`。

```

1 void ConsumerController::start() {
2     // TODO: starts a ConsumerController thread
3     if (rec_ptr) {
4         rec_ptr->clear();
5         rec_ptr->push_back(0);
6     }
7     global_ctr = this;
8     pthread_create(&t, nullptr, ConsumerController::process, (void
9 *) this);
9 }

```

ConsumerController::process

為了完成在一定時間內檢查並對目前擁有的 `Consumer` 數量進行 scaling, 我選擇使用 `sigaction` 和 `timer` 這兩個 Unix API 來完成。

參考 [How to setup a "precise" periodic timer to monitor stuff in Linux\(C/C++\)?](#) 的介紹, 我...

1. 建立一個 `sigaction` 物件, 設定 `struct sigaction::sa_handler` 成員 (實際上這是一個 macro, 會展開為正確的內部成員) 為 `static ConsumerController::handler` 靜態方法, 最後呼叫 `sigaction` 函式, 將此物件的參考傳入, 同時設定要觸發的 signal 類型, 在此我選擇一個簡單的 `SIGALRM` (signal alarm)。
2. 設定 timer, 先建立 `itimerval` 物件方便 Linux 幫我們計時, 初始化其 `itimerval::it_interval` (表計時歸零後要重新初始化的值) 和 `itimerval::it_value` (表計時器一開始的數值)。最後調用 `setitimer` 函式開始計時, 傳入剛才創建好的 `itimerval` 物件, 並設定計時種類為 `ITIMER_REAL` 表示以系統時間為準。
3. 最後進入無限迴圈, 等待 `ConsumerController::cancel` 被呼叫使 `global_ctr` 變為 `nullptr` 後結束回圈, 並回傳空指標。

故整體程式碼可以實作如下:

```

1 void* ConsumerController::process(void* arg) {
2     // TODO: implements the ConsumerController's work
3     ConsumerController * cur = (ConsumerController *) arg;

```

```

4
5     // signal action alarm handler
6     struct sigaction act; // make a signal action
7     act.sa_handler = handler; // assign handler
8     sigaction(SIGALRM, &act, nullptr); // set signal triggering
9
10    // set periodic timer
11    long u_period = cur->check_period; // interval in microsecond
12    itimerval itv; // interval timer
13    itv.it_interval = itv.it_value = { u_period / 1000000,
u_period % 1000000 }; // init timer value
14    setitimer(ITIMER_REAL, &itv, nullptr); // set timer w.r.t.
real system time
15
16    // infinite loop
17    while (global_ctr);
18
19    return nullptr;
20 }

```

ConsumerController::handler

最後是我新增的 `ConsumerController::handler` 方法, 其在每次計時器歸零時都會被呼叫, 傳入訊號觸發的類型 `signo`。故在確定 signal 為方才設定好的 `SIGALRM` 後, 便能開始 `Consumer` 數量的監控與調整邏輯:

1. worker size 超過上界: 增加一個 `Consumer`
2. worker size 低於下界且不為一: 減少一個 `Consumer`

故整體程式碼實作為:

```

1 void ConsumerController::handler(int signo) {
2     if (global_ctr) {
3         int wk_sz = global_ctr->worker_queue->get_size(), // size
of work_queue
4         cm_sz = global_ctr->consumers.size(); // size of
consumer

```

```

5         switch (signo) {
6             case SIGALRM:
7                 if (wk_sz > global_ctr->high_threshold) {
8                     global_ctr->consumers.push_back(new
Consumer(global_ctr->worker_queue, global_ctr->writer_queue,
global_ctr->transformer));
9                     global_ctr->consumers.back()->start();
10                    if (print_scaling_msg)
11                        std::cout << "Scaling up consumers from " <<
cm_sz << " to " << (cm_sz + 1) << std::endl;
12                    if (rec_ptr)
13                        rec_ptr->push_back(cm_sz + 1);
14                }
15                else if (wk_sz < global_ctr->low_threshold && cm_sz >
1) {
16                    global_ctr->consumers.back()->cancel();
17                    global_ctr->consumers.pop_back();
18                    if (print_scaling_msg)
19                        std::cout << "Scaling down consumers from " <<
cm_sz << " to " << (cm_sz - 1) << std::endl;
20                    if (rec_ptr)
21                        rec_ptr->push_back(cm_sz + 1);
22                }
23                break;
24            }
25        }
26    }

```

至此, 除 main 函式以外的邏輯都實作完畢。

implementation of main

我為主程式設計了兩個幫手函式: `one_run` 和 `one_experiment`, 他們的目的是:

1. `one_run`: 根據我們想要的參數進行一次運行, 傳入一次正常執行所需要的全部參數資訊, 具體的參數在本函式的簽名寫得很清楚, 故不贅述。
2. `one_experiment`: 進行一次「實驗」, 其會調用 `one_run` 函式, 並計時與寫出實驗

數據。

以下讓我介紹這兩個函式。

one_run

根據傳入的參數初始化 `input_queue`, `worker_queue`, `writer_queue`, `reader`, `writer`, `transformer`, `pv` (vector of producer), `ctr` (consumer controller) 這些執行緒, 並執行之, 然後等待 `reader` 和 `writer` 完成後把這些執行緒刪除。

```
1  /**
2   * Conduct a single run with assigned parameters
3   * @param r_qs      reader queue size
4   * @param wk_qs     worker queue size
5   * @param wt_qs     writer queue size
6   * @param cs_h_th   consumer controller high threshold size
7   * @param cs_l_th   consumer controller low threshold size
8   * @param cs_per    check period of consumer controller
9   * @param in        input file name
10  * @param out        output file name
11  * @param lines      numbers of lines in "in" file
12  */
13 void one_run(
14     int r_qs, int wk_qs, int wt_qs,
15     int cs_h_th, int cs_l_th, int cs_per,
16     std::string & in, std::string & out,
17     int lines)
18 {
19     // prepare queues
20     auto input_queue = new TSQueue<Item *>(r_qs),
21         worker_queue = new TSQueue<Item *>(wk_qs),
22         writer_queue = new TSQueue<Item *>(wt_qs);
23
24     // init and start reader thread
25     auto reader = new Reader(lines, in, input_queue);
26     reader->start();
27 }
```



```
28     // init and start writer thread
29     auto writer = new Writer(lines, out, writer_queue);
30     writer->start();
31
32     // prepare transformer
33     auto transformer = new Transformer();
34
35     // producer vector, init with four identical producers and
36     start all of them
37     std::vector<Producer> pv(4, Producer(input_queue,
38     worker_queue, transformer));
39     for (auto &p: pv) p.start(); // start all producers
40
41     // init and start consumer controller thread
42     auto ctr = new ConsumerController(
43     worker_queue, writer_queue, transformer,
44     cs_per, wk_qs * cs_h_th / 100, wk_qs * cs_l_th / 100);
45     ctr->start();
46
47     // join reader and writer threads
48     reader->join();
49     writer->join();
50
51     // delete all threads
52     delete reader;
53     delete writer;
54     delete ctr;
55     delete transformer;
56
57     // delete all queue
58     delete input_queue;
59     delete worker_queue;
60     delete writer_queue;
61 }
```

one_experiment

為了完成實驗的隔離性與紀錄功能, `one_experiment` 要完成以下三件事:

1. `fork` 出新的 process 以完全隔絕每次實驗的數據, 使他們不互相影響, 會 wait 至實驗結束。
2. 實驗前後會設置計時器, 紀錄實驗花費的時間
3. 如果有指定要寫入實驗數據的 `log` 檔檔名, 就會將計時的節果和 `ConsumerController` scaling 的結果寫入。

故實作可以為以下, 注意到為了精確的計時, 我使用 C++ `std::chrono` 的功能來完成。

```
1  /**
2   * Conduct a single experiment with assigned parameters
3   * @param r_qs      reader queue size
4   * @param wk_qs     worker queue size
5   * @param wt_qs     writer queue size
6   * @param cs_h_th   consumer controller high threshold size
7   * @param cs_l_th   consumer controller low threshold size
8   * @param cs_per    check period of consumer controller
9   * @param in        input file name
10  * @param out        output file name
11  * @param log        log file name (pointer, since this is optional)
12  * @param lines      numbers of lines in "in" file
13  * @param to_print   whether to print the experiment information
14  */
15 void one_experiment(
16     int r_qs, int wk_qs, int wt_qs,
17     int cs_h_th, int cs_l_th, int cs_per,
18     std::string & in, std::string & out, std::string * log,
19     int lines, bool to_print)
20 {
21     int pid = fork();
22
23     if (!pid) {
24         // is child process
25         // experiment timer start
26         auto start = std::chrono::high_resolution_clock::now();
```

```

27
28     // start the experience
29     one_run(
30         r_qs, wk_qs, wt_qs,
31         cs_h_th, cs_l_th, cs_per,
32         in, out, lines);
33
34     // experiment timer end
35     auto end = std::chrono::high_resolution_clock::now();
36
37     // calculate time consumption
38     double result =
39         std::chrono::duration_cast<std::chrono::nanoseconds>(end -
40         start).count();
41     result *= 1e-9;
42
43     if (to_print) {
44         // print experiment message
45         std::cout << "\nQueue size:\n    (reader, worker,
46         writer) = ("
47             << r_qs << ", " << wk_qs << ", " << wt_qs << ")"
48         << std::endl;
49         std::cout << "Consumer settings:\n    (high, low,
50         period) = ("
51             << cs_h_th << "%, " << cs_l_th << "%, ";
52         if (cs_per > 1000000)
53             std::cout << (cs_per / 1000000.) << "s)" <<
54             std::endl;
55         else
56             std::cout << (cs_per / 1000.) << "ms)" <<
57             std::endl;
58         // print time consumption
59         std::cout << "Total time usage: "
60             << result << std::setprecision(6)
61             << "s" << std::endl << std::endl;
62     }
63
64     // write experiment message to log file
65     if (log) {

```

```

59         std::ofstream log_file(*log, std::ios::app);
60         log_file << result << " / [";
61         for (auto n: *rec_ptr)
62             log_file << n << ", ";
63         log_file << "]" << std::endl;
64         log_file.close();
65     }
66
67     exit(0); // end of child process
68 }
69 // wait for child process to end
70 wait(nullptr);
71 }

```

main

最後, `main` 函式負責根據執行時的參數反覆呼叫 `one_experiment`, 完成所要求的執行內容。

注意到我這樣的設計使我們不需用修改 `macro` 的數值, 而可以非常客製化的調整實驗的參數與環境。另外, 執行時參數的解析我使用 `unordered_set` 讓流程變得輕鬆寫意:

```

1  int main(int argc, char** argv) {
2      // TODO: implements main function
3      // parse the parameters
4      std::unordered_set<std::string> args; // arguments
5      for (int i = 0; i < argc; ++i)
6          args.insert(argv[i]);
7
8      // identify the assigned functions
9      bool to_exp = args.find("expe") != args.end();
10     bool to_debug = args.find("msg") != args.end();
11
12     // check if all the arguments are valid
13     bool all_args_valid = to_exp + to_debug + 4 == argc;
14
15     // check legalness of arguments

```

```

16     assert(all_args_valid);
17
18     int n = atoi(argv[1]); // expected lines
19     std::string input_file_name(argv[2]);
20     std::string output_file_name(argv[3]);
21
22     // rename macros for better experiment convenience
23     int R_QS  = READER_QUEUE_SIZE, // reader queue size
24         WK_QS = WORKER_QUEUE_SIZE, // worker queue size
25         WT_QS = WRITER_QUEUE_SIZE, // writer queue size
26         CS_H_TH = CONSUMER_CONTROLLER_HIGH_THRESHOLD_PERCENTAGE,
27         CS_L_TH = CONSUMER_CONTROLLER_LOW_THRESHOLD_PERCENTAGE,
28         CS_PER  = CONSUMER_CONTROLLER_CHECK_PERIOD; // consumer
// consumer controller high threshold percentage
// consumer controller low threshold percentage
// consumer controller check period
29
30     // basic run
31     one_experiment(
32         R_QS, WK_QS, WT_QS,
33         CS_H_TH, CS_L_TH, CS_PER,
34         input_file_name, output_file_name,
35         nullptr, n, false);
36
37     // conduct all experiments
38     if (to_exp) {
39         // variables for experience
40         const int expe_times = 8;
41         double usage;
42         std::string out_file("./tmp/tmp.out"); // dami output
file
43
44         // assign global variables for experiment
45         rec_ptr = new std::vector<int>();
46         print_scaling_msg = false;
47
48         // experiences
49         {

```

```

50         std::cout << "----- [start of period test] ----
-----" << std::endl;
51         std::string log_name("./report/log/period.log");
52         std::ofstream log_file(log_name);
53         log_file << "Period / Time Usage / Consumer Size
Change List" << std::endl;
54         log_file.close();
55         for (long long i = 1 << 10; i <= 1 << 24; i <= 2) {
56             for (int j = 0; j < expe_times; ++j) {
57                 log_file.open(log_name, std::ios::app);
58                 log_file << i << " / ";
59                 log_file.close();
60                 one_experiment(
61                     R_QS, WK_QS, WT_QS,
62                     CS_H_TH, CS_L_TH, i,
63                     input_file_name, out_file,
64                     &log_name, n, to_debug);
65             }
66         }
67         std::cout << "----- [end of period test] ----
-----" << std::endl << std::endl;
68     }
69
70     {
71         std::cout << "----- [start of low threshold test]
-----" << std::endl;
72         std::string
log_name("./report/log/low_threshold.log");
73         std::ofstream log_file(log_name);
74         log_file << "Low Threshold / Time Usage / Consumer
Size Change List" << std::endl;
75         log_file.close();
76         for (int i = 0; i < CS_H_TH; i += 5) {
77             for (int j = 0; j < expe_times; ++j) {
78                 log_file.open(log_name, std::ios::app);
79                 log_file << i << " / ";
80                 log_file.close();
81                 one_experiment(
82                     R_QS, WK_QS, WT_QS,

```

```

83         CS_H_TH, i, CS_PER,
84         input_file_name, out_file,
85         &log_name, n, to_debug);
86     }
87 }
88     std::cout << "----- [end of low threshold test] -
-----" << std::endl << std::endl;
89 }
90
91 {
92     std::cout << "----- [start of high threshold test]
-----" << std::endl;
93     std::string
log_name("./report/log/high_threshold.log");
94     std::ofstream log_file(log_name);
95     log_file << "High Threshold / Time Usage / Consumer
Size Change List" << std::endl;
96     log_file.close();
97     for (int i = CS_L_TH + 5; i <= 100; i += 5) {
98         for (int j = 0; j < expe_times; ++j) {
99             log_file.open(log_name, std::ios::app);
100             log_file << i << " / ";
101             log_file.close();
102             one_experiment(
103                 R_QS, WK_QS, WT_QS,
104                 i, CS_L_TH, CS_PER,
105                 input_file_name, out_file,
106                 &log_name, n, to_debug);
107         }
108     }
109     std::cout << "----- [end of high threshold test] -
-----" << std::endl << std::endl;
110 }
111
112 {
113     std::cout << "----- [start of worker queue test] -
-----" << std::endl;
114     std::string log_name("./report/log/work_size.log");
115     std::ofstream log_file(log_name);

```

```

116         log_file << "Worker Queue Size / Time Usage /
Consumer Size Change List" << std::endl;
117         log_file.close();
118         for (int i = 20; i <= 400; i += 20) {
119             for (int j = 0; j < expe_times; ++j) {
120                 log_file.open(log_name, std::ios::app);
121                 log_file << i << " / ";
122                 log_file.close();
123                 one_experiment(
124                     R_QS, i, WT_QS,
125                     CS_H_TH, CS_L_TH, CS_PER,
126                     input_file_name, out_file,
127                     &log_name, n, to_debug);
128             }
129         }
130         std::cout << "----- [end of worker queue test] --
-----" << std::endl << std::endl;
131     }
132
133     {
134         std::cout << "----- [start of writer queue test] -
-----" << std::endl;
135         std::string log_name("./report/log/write_size.log");
136         std::ofstream log_file(log_name);
137         log_file << "Writer Queue Size / Time Usage /
Consumer Size Change List" << std::endl;
138         log_file.close();
139         for (int i = 250; i <= 7500; i += 250) {
140             for (int j = 0; j < expe_times; ++j) {
141                 log_file.open(log_name, std::ios::app);
142                 log_file << i << " / ";
143                 log_file.close();
144                 one_experiment(
145                     R_QS, WK_QS, i,
146                     CS_H_TH, CS_L_TH, CS_PER,
147                     input_file_name, out_file,
148                     &log_name, n, to_debug);
149             }
150         }

```



```

151         std::cout << "----- [end of writer queue test] --
-----" << std::endl << std::endl;
152     }
153
154     {
155         std::cout << "----- [start of reader queue test] -
-----" << std::endl;
156         std::string log_name("./report/log/read_size.log");
157         std::ofstream log_file(log_name);
158         log_file << "Reader Queue Size / Time Usage /
Consumer Size Change List" << std::endl;
159         log_file.close();
160         for (int i = 5; i <= 400; i += 5) {
161             for (int j = 0; j < expe_times; ++j) {
162                 log_file.open(log_name, std::ios::app);
163                 log_file << i << " / ";
164                 log_file.close();
165                 one_experiment(
166                     i, WK_QS, WT_QS,
167                     CS_H_TH, CS_L_TH, CS_PER,
168                     input_file_name, out_file,
169                     &log_name, n, to_debug);
170             }
171         }
172         std::cout << "----- [end of reader queue test] --
-----" << std::endl << std::endl;
173     }
174
175     delete rec_ptr;
176 }
177
178     return 0;
179 }

```

如此一來,所有實作都完成了,接著來討論實驗的結果。

Experiment

實驗部分, 每一個 part 我們只會改變一個變數, 藉此來觀察該變數對於 Time Usage 與 Consumer Size Change List 的影響, 每次實驗我們都做 8 次, 最後將值取平均, 另外我們發現 Consumer Size Change List 的長度會不規則抖動, 推測和 sever 有關, 對此我們只會採 8 次中長度相同占比最多的數據

Different values of CONSUMER_CONTROLLER_CHECK_PERIOD

► CONSUMER_CONTROLLER_CHECK_PERIOD 由小到大產生的 data

```
1  ['Period', 'Time Usage', 'Consumer Size Change List']
2  [[1024.0 52.815733333333334
3    array([0.          , 1.          , 2.          , ..., 3.          ,
4          2.66666667,
5          3.          ])]
6    [4096.0 52.8669 array([0., 1., 2., ..., 3., 2., 3.])]
7    [16384.0 52.5369 array([0., 1., 2., ..., 5., 4., 3.])]
8    [65536.0 51.1949
9    array([ 0.,  1.,  2.,  3.,  4.,  5.,  6.,  7.,  8.,  9., 10.,
10         11., 12.,
11         13., 14., 15., 14., 13., 12., 11., 10.,  9.,  8.,  7.,
12         6.,  5.,
13         4.,  3.,  4.,  5.,  6.,  7.,  8.,  9., 10., 11., 12.,
14         13., 14.,
15         15., 14., 13., 12., 11., 10.,  9.,  8.,  7.,  6.,  5.,
16         4.,  3.,
17         4.,  5.,  6.,  7.,  8.,  9., 10., 11., 12., 13., 14.,
18         13., 12.,
19         11., 10.,  9.,  8.,  7.,  6.,  5.,  4.,  3.,  4.,  5.,
20         6.,  7.,
21         8.,  9., 10., 11., 12., 11., 10.,  9.,  8.,  7.,  6.,
22         5.,  4.,
```

[illegible]

[illegible]

```

55         5., 4., 5., 6., 7., 8., 9., 10., 11., 12., 13.,
12., 11.,
56         10., 9., 8., 7., 6., 7., 8., 9., 10., 11., 10.,
9., 8.,
57         7., 6., 7., 8., 9., 10., 11., 12., 11., 10., 9.,
8., 7.,
58         6., 7., 8., 9., 10., 11., 12., 11., 10., 9., 8.,
7., 6.,
59         5., 6., 7., 8., 9., 10., 11., 12., 11., 10., 9.,
8., 7.,
60         6., 5., 4., 3.]))
        ]
61 [262144.0 51.7688
62  array([ 0.          ,  1.          ,  2.          ,  3.          ,  4.
63         ,
64         5.          ,  6.          ,  7.          ,  8.          ,  9.
65         ,
66         10.         , 11.          , 12.          , 13.          , 14.
67         ,
68         15.         , 16.          , 17.          , 16.          , 15.
69         ,
70         14.         , 13.          , 12.          , 11.          , 10.
71         ,
72         9.          , 8.           , 7.           , 6.           , 5.
73         ,
74         4.          , 3.          , 2.          , 3.          , 4.
75         ,
76         3.          , 2.          , 3.          , 4.          , 3.
77         ,
78         2.          , 3.          , 4.          , 3.          , 2.
79         ,
80         3.          , 4.          , 3.          , 2.          , 3.
81         ,
82         4.          , 3.          , 2.          , 3.          , 4.
83         ,
84         3.          , 2.          , 3.          , 2.          , 3.
85         ,
86         4.          , 3.          , 2.          , 3.          , 4.
87         ,
88         3.          , 2.          , 3.          , 2.          , 3.
89         ,
90         4.          , 3.          , 2.          , 3.          , 4.
91         ,
92         3.          , 2.          , 3.          , 2.          , 3.
93         ,
94         4.          , 3.          , 2.          , 3.          , 4.
95         ,
96         3.          , 2.          , 3.          , 2.          , 3.
97         ,
98         4.          , 3.          , 2.          , 3.          , 4.
99         ,
100        3.          , 2.          , 3.          , 2.          , 3.
101        ,
102        4.          , 3.          , 2.          , 3.          , 4.
103        ,
104        3.          , 2.          , 3.          , 2.          , 3.
105        ,
106        4.          , 3.          , 2.          , 3.          , 4.
107        ,
108        3.          , 2.          , 3.          , 2.          , 3.
109        ,
110        4.          , 3.          , 2.          , 3.          , 4.
111        ,
112        3.          , 2.          , 3.          , 2.          , 3.
113        ,
114        4.          , 3.          , 2.          , 3.          , 4.
115        ,
116        3.          , 2.          , 3.          , 2.          , 3.
117        ,
118        4.          , 3.          , 2.          , 3.          , 4.
119        ,
120        3.          , 2.          , 3.          , 2.          , 3.
121        ,
122        4.          , 3.          , 2.          , 3.          , 4.
123        ,
124        3.          , 2.          , 3.          , 2.          , 3.
125        ,
126        4.          , 3.          , 2.          , 3.          , 4.
127        ,
128        3.          , 2.          , 3.          , 2.          , 3.
129        ,
130        4.          , 3.          , 2.          , 3.          , 4.
131        ,
132        3.          , 2.          , 3.          , 2.          , 3.
133        ,
134        4.          , 3.          , 2.          , 3.          , 4.
135        ,
136        3.          , 2.          , 3.          , 2.          , 3.
137        ,
138        4.          , 3.          , 2.          , 3.          , 4.
139        ,
140        3.          , 2.          , 3.          , 2.          , 3.
141        ,
142        4.          , 3.          , 2.          , 3.          , 4.
143        ,
144        3.          , 2.          , 3.          , 2.          , 3.
145        ,
146        4.          , 3.          , 2.          , 3.          , 4.
147        ,
148        3.          , 2.          , 3.          , 2.          , 3.
149        ,
150        4.          , 3.          , 2.          , 3.          , 4.
151        ,
152        3.          , 2.          , 3.          , 2.          , 3.
153        ,
154        4.          , 3.          , 2.          , 3.          , 4.
155        ,
156        3.          , 2.          , 3.          , 2.          , 3.
157        ,
158        4.          , 3.          , 2.          , 3.          , 4.
159        ,
160        3.          , 2.          , 3.          , 2.          , 3.
161        ,
162        4.          , 3.          , 2.          , 3.          , 4.
163        ,
164        3.          , 2.          , 3.          , 2.          , 3.
165        ,
166        4.          , 3.          , 2.          , 3.          , 4.
167        ,
168        3.          , 2.          , 3.          , 2.          , 3.
169        ,
170        4.          , 3.          , 2.          , 3.          , 4.
171        ,
172        3.          , 2.          , 3.          , 2.          , 3.
173        ,
174        4.          , 3.          , 2.          , 3.          , 4.
175        ,
176        3.          , 2.          , 3.          , 2.          , 3.
177        ,
178        4.          , 3.          , 2.          , 3.          , 4.
179        ,
180        3.          , 2.          , 3.          , 2.          , 3.
181        ,
182        4.          , 3.          , 2.          , 3.          , 4.
183        ,
184        3.          , 2.          , 3.          , 2.          , 3.
185        ,
186        4.          , 3.          , 2.          , 3.          , 4.
187        ,
188        3.          , 2.          , 3.          , 2.          , 3.
189        ,
190        4.          , 3.          , 2.          , 3.          , 4.
191        ,
192        3.          , 2.          , 3.          , 2.          , 3.
193        ,
194        4.          , 3.          , 2.          , 3.          , 4.
195        ,
196        3.          , 2.          , 3.          , 2.          , 3.
197        ,
198        4.          , 3.          , 2.          , 3.          , 4.
199        ,
200        3.          , 2.          , 3.          , 2.          , 3.
201        ,
202        4.          , 3.          , 2.          , 3.          , 4.
203        ,
204        3.          , 2.          , 3.          , 2.          , 3.
205        ,
206        4.          , 3.          , 2.          , 3.          , 4.
207        ,
208        3.          , 2.          , 3.          , 2.          , 3.
209        ,
210        4.          , 3.          , 2.          , 3.          , 4.
211        ,
212        3.          , 2.          , 3.          , 2.          , 3.
213        ,
214        4.          , 3.          , 2.          , 3.          , 4.
215        ,
216        3.          , 2.          , 3.          , 2.          , 3.
217        ,
218        4.          , 3.          , 2.          , 3.          , 4.
219        ,
220        3.          , 2.          , 3.          , 2.          , 3.
221        ,
222        4.          , 3.          , 2.          , 3.          , 4.
223        ,
224        3.          , 2.          , 3.          , 2.          , 3.
225        ,
226        4.          , 3.          , 2.          , 3.          , 4.
227        ,
228        3.          , 2.          , 3.          , 2.          , 3.
229        ,
230        4.          , 3.          , 2.          , 3.          , 4.
231        ,
232        3.          , 2.          , 3.          , 2.          , 3.
233        ,
234        4.          , 3.          , 2.          , 3.          , 4.
235        ,
236        3.          , 2.          , 3.          , 2.          , 3.
237        ,
238        4.          , 3.          , 2.          , 3.          , 4.
239        ,
240        3.          , 2.          , 3.          , 2.          , 3.
241        ,
242        4.          , 3.          , 2.          , 3.          , 4.
243        ,
244        3.          , 2.          , 3.          , 2.          , 3.
245        ,
246        4.          , 3.          , 2.          , 3.          , 4.
247        ,
248        3.          , 2.          , 3.          , 2.          , 3.
249        ,
250        4.          , 3.          , 2.          , 3.          , 4.
251        ,
252        3.          , 2.          , 3.          , 2.          , 3.
253        ,
254        4.          , 3.          , 2.          , 3.          , 4.
255        ,
256        3.          , 2.          , 3.          , 2.          , 3.
257        ,
258        4.          , 3.          , 2.          , 3.          , 4.
259        ,
260        3.          , 2.          , 3.          , 2.          , 3.
261        ,
262        4.          , 3.          , 2.          , 3.          , 4.
263        ,
264        3.          , 2.          , 3.          , 2.          , 3.
265        ,
266        4.          , 3.          , 2.          , 3.          , 4.
267        ,
268        3.          , 2.          , 3.          , 2.          , 3.
269        ,
270        4.          , 3.          , 2.          , 3.          , 4.
271        ,
272        3.          , 2.          , 3.          , 2.          , 3.
273        ,
274        4.          , 3.          , 2.          , 3.          , 4.
275        ,
276        3.          , 2.          , 3.          , 2.          , 3.
277        ,
278        4.          , 3.          , 2.          , 3.          , 4.
279        ,
280        3.          , 2.          , 3.          , 2.          , 3.
281        ,
282        4.          , 3.          , 2.          , 3.          , 4.
283        ,
284        3.          , 2.          , 3.          , 2.          , 3.
285        ,
286        4.          , 3.          , 2.         
```

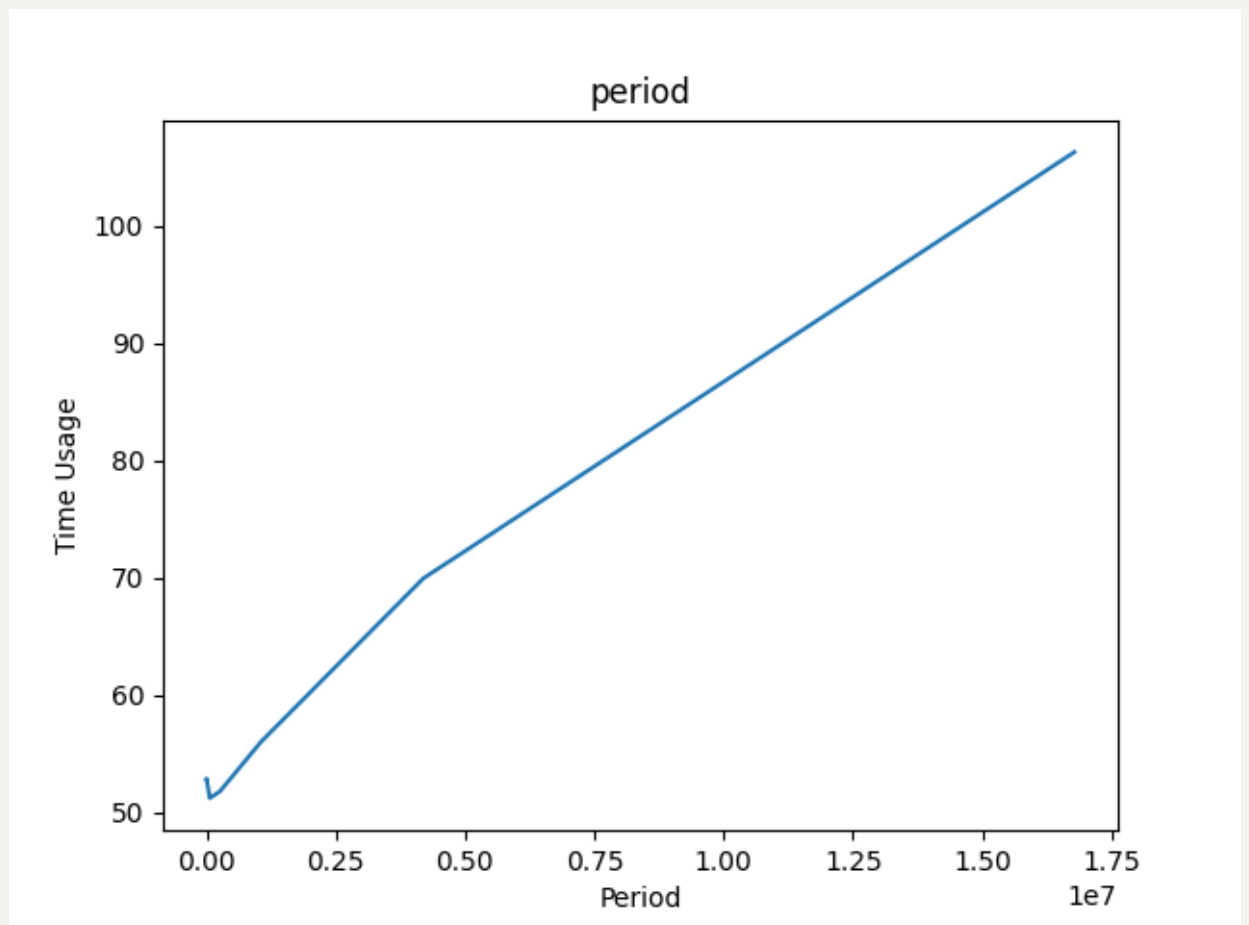
75	3.	,	2.	,	3.	,	4.	,	3.
	,								
76	2.	,	3.	,	4.	,	3.	,	2.
	,								
77	3.	,	4.	,	3.	,	4.	,	5.
	,								
78	6.	,	7.	,	8.	,	9.	,	10.
	,								
79	11.	,	12.	,	13.	,	14.	,	13.
	,								
80	12.	,	11.	,	10.	,	9.	,	8.
	,								
81	7.	,	6.	,	5.	,	4.	,	5.
	,								
82	6.	,	7.	,	8.	,	9.	,	10.
	,								
83	11.	,	12.	,	11.	,	10.	,	9.
	,								
84	8.	,	7.	,	6.	,	5.	,	4.
	,								
85	3.	,	2.	,	3.	,	4.	,	3.
	,								
86	2.	,	3.	,	4.	,	3.	,	2.
	,								
87	3.	,	2.	,	3.	,	4.	,	3.
	,								
88	2.	,	3.	,	4.	,	3.	,	2.
	,								
89	3.	,	4.	,	3.	,	2.	,	3.
	,								
90	4.	,	3.	,	2.	,	3.	,	4.
	,								
91	3.	,	2.	,	3.	,	3.33333333	,	3.
	,								
92	2.66666667	,	3.	,	4.	,	3.	,	2.
	,								
93	3.	,	4.	,	5.	,	6.	,	7.
	,								

```

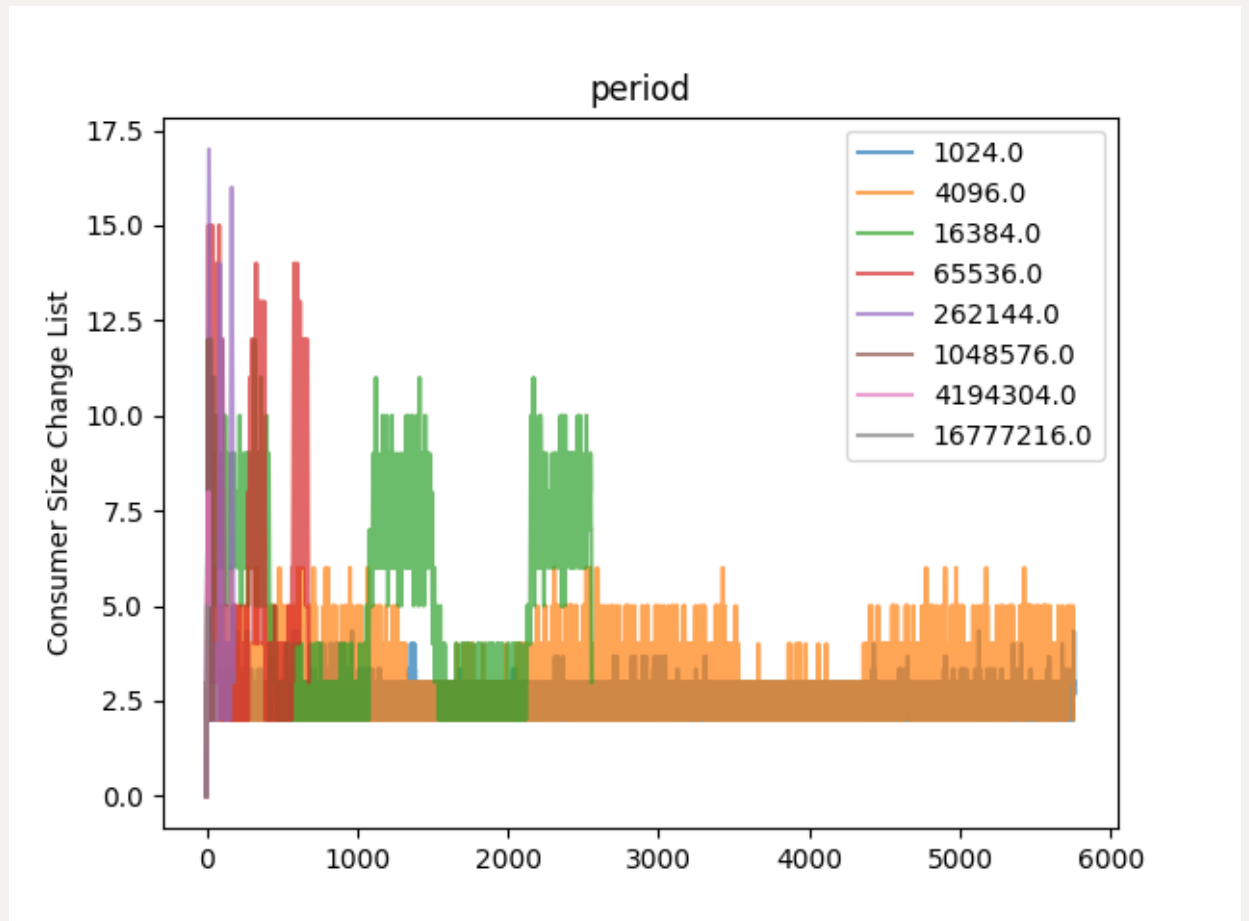
94         8.         , 9.         , 10.         , 11.         , 12.
95         ,
96         13.         , 14.         , 15.         , 16.         , 15.
97         ,
98         14.         , 13.         , 12.         , 11.         , 10.
99         ,
100        9.         , 8.         , 7.         , 6.         , 5.
101        ,
102        4.         , 5.         , 4.         , 3.         ] )
103    ]
104    [1048576.0 55.9612
105    array([ 0.,  1.,  2.,  3.,  4.,  5.,  6.,  7.,  8.,  9., 10.,
106          11., 12.,
107          11., 10.,  9.,  8.,  7.,  6.,  5.,  4.,  3.,  2.,  3.,
108          4.,  5.,
109          6.,  7.,  8.,  9., 10., 11., 12., 11., 10.,  9.,  8.,
110          7.,  6.,
111          5.,  4.,  3.,  2.,  3.,  4.,  5.,  6.,  7.,  8.,  9.,
112          10., 11.])]
113    [4194304.0 69.938125
114    array([0.,  1.,  2.,  3.,  4.,  5.,  6.,  7.,  6.,  5.,  6.,  7.,  8.,  7.,
115          6.,  5.,  6.])]
116    [16777216.0 106.28974999999998 array([0.,  1.,  2.,  3.,  4.,  5.,
117          4.])]]

```

1. 圖像化 PERIOD 對 Time Usage 的影響:



2. 圖像化 PERIOD 對 CONSUMER SIZE CHANGE LIST 的影響:



3. 討論:

- 從上圖可以明顯發現 Time Usage 和 period 大致呈正相關, period 越大, Time Usage 就越久
- 另外可以發現, 當 period 越小, CONSUMER SIZE CHANGE 的次數就越多, 但幅度較小, 反觀是 period 小的時候, 改變幅度大, 有明顯高峰與低谷, 且改變次數大幅減少

Different values of CONSUMER_CONTROLLER_HIGH/LOW_THRESHOLD_PERCENTAGE

► CONSUMER_CONTROLLER_LOW_THRESHOLD_PERCENTAGE 由小到大產生的 data

```

1  ['Low Threshold', 'Time Usage', 'Consumer Size Change List']
2  [[0.0 55.35764285714286
3    array([ 0.,  1.,  2.,  3.,  4.,  5.,  6.,  7.,  8.,  9., 10.,
4    11., 12.,
5          11., 10.,  9.,  8.,  7.,  6.,  5.,  4.,  3.,  4.,  3.,
6    4.,  3.,
7          4.,  5.,  6.,  7.,  8.,  9., 10., 11., 12., 11., 10.,
8    9.,  8.,
9          7.,  6.,  5.,  4.,  3.,  4.,  3.,  4.,  3.,  4.,  5.,
10   6.,  7.,
11          8.,  9., 10., 11.]])
12   ]
13   [5.0 55.355150000000001
14   array([ 0.,  1.,  2.,  3.,  4.,  5.,  6.,  7.,  8.,  9., 10.,
15   11., 12.,
16          11., 10.,  9.,  8.,  7.,  6.,  5.,  4.,  3.,  2.,  3.,
17   4.,  3.,
18          4.,  5.,  6.,  7.,  8.,  9., 10., 11., 12., 11., 10.,
19   9.,  8.,
20          7.,  6.,  5.,  4.,  3.,  2.,  3.,  4.,  3.,  4.,  5.,
21   6.,  7.,
22          8.,  9., 10., 11.]])
23   ]
24   [10.0 55.358062499999996
25   array([ 0.,  1.,  2.,  3.,  4.,  5.,  6.,  7.,  8.,  9., 10.,
26   11., 12.,

```

```

16         11., 10., 9., 8., 7., 6., 5., 4., 3., 2., 3.,
    4., 3.,
17         4., 5., 6., 7., 8., 9., 10., 11., 12., 11., 10.,
    9., 8.,
18         7., 6., 5., 4., 3., 2., 3., 4., 3., 4., 5.,
    6., 7.,
19         8., 9., 10., 11.]))
    ]
20 [15.0 53.8285250000000006
21  array([ 0., 1., 2., 3., 4., 5., 6., 7., 8., 9., 10.,
    11., 12.,
22         11., 10., 9., 8., 7., 6., 5., 4., 3., 2., 3.,
    4., 5.,
23         6., 7., 8., 9., 10., 11., 12., 11., 10., 9., 8.,
    7., 6.,
24         5., 4., 3., 2., 3., 4., 5., 6., 7., 8., 9.,
    10., 11.)))]
25 [20.0 55.354971428571424
26  array([ 0., 1., 2., 3., 4., 5., 6., 7., 8., 9., 10.,
    11., 12.,
27         11., 10., 9., 8., 7., 6., 5., 4., 3., 2., 3.,
    4., 5.,
28         6., 7., 8., 9., 10., 11., 12., 11., 10., 9., 8.,
    7., 6.,
29         5., 4., 3., 2., 3., 4., 3., 4., 5., 6., 7.,
    8., 9.,
30         10., 11.]))
    ]
31 [25.0 55.413512499999996
32  array([ 0., 1., 2., 3., 4., 5., 6., 7., 8., 9., 10.,
    11., 12.,
33         11., 10., 9., 8., 7., 6., 5., 4., 3., 2., 3.,
    4., 5.,
34         6., 7., 8., 9., 10., 11., 12., 11., 10., 9., 8.,
    7., 6.,
35         5., 4., 3., 2., 3., 4., 3., 4., 5., 6., 7.,
    8., 9.,
36         10., 11.]))
    ]

```

```

37 [30.0 55.7284
38   array([ 0.,  1.,  2.,  3.,  4.,  5.,  6.,  7.,  8.,  9., 10.,
39          11., 12.,
          11., 10.,  9.,  8.,  7.,  6.,  5.,  4.,  3.,  2.,  3.,
40          4.,  5.,
          6.,  7.,  8.,  9., 10., 11., 12., 11., 10.,  9.,  8.,
41          7.,  6.,
          5.,  4.,  3.,  2.,  3.,  4.,  5.,  6.,  7.,  8.,  9.,
42          10., 11.]])
43 [35.0 55.7040875
44   array([ 0.,  1.,  2.,  3.,  4.,  5.,  6.,  7.,  8.,  9., 10.,
45          11., 12.,
          11., 10.,  9.,  8.,  7.,  6.,  5.,  4.,  3.,  2.,  3.,
46          4.,  5.,
          6.,  7.,  8.,  9., 10., 11., 12., 11., 10.,  9.,  8.,
          7.,  6.,
          5.,  4.,  3.,  2.,  3.,  4.,  5.,  6.,  7.,  8.,  9.,
47          10., 11.]])
48 [40.0 55.706537499999996
49   array([ 0.,  1.,  2.,  3.,  4.,  5.,  6.,  7.,  8.,  9., 10.,
50          11., 12.,
          11., 10.,  9.,  8.,  7.,  6.,  5.,  4.,  3.,  2.,  3.,
          4.,  5.,
          6.,  7.,  8.,  9., 10., 11., 12., 11., 10.,  9.,  8.,
          7.,  6.,
          5.,  4.,  3.,  2.,  3.,  4.,  5.,  6.,  7.,  8.,  9.,
51          10., 11.]])
52 [45.0 55.68725
53   array([ 0.,  1.,  2.,  3.,  4.,  5.,  6.,  7.,  8.,  9., 10.,
54          11., 10.,
          9.,  8.,  7.,  6.,  5.,  4.,  3.,  2.,  3.,  4.,  5.,
          6.,  7.,
          8.,  9., 10., 11., 12., 11., 10.,  9.,  8.,  7.,  6.,
          5.,  4.,
          3.,  2.,  3.,  4.,  5.,  6.,  7.,  8.,  9., 10., 11.]])
55   ]
56 [50.0 55.823749999999999
57   array([ 0.,  1.,  2.,  3.,  4.,  5.,  6.,  7.,  8.,  9., 10.,
58          11., 10.,

```

```

59         9., 8., 7., 6., 5., 4., 3., 2., 3., 4., 5.,
        6., 7.,
60         8., 9., 10., 11., 12., 11., 10., 9., 8., 7., 6.,
        5., 4.,
61         3., 2., 3., 4., 5., 6., 7., 8., 9., 10., 11.])
        ]
62 [55.0 56.8237
63   array([ 0.,  1.,  2.,  3.,  4.,  5.,  6.,  7.,  8.,  9., 10.,
64          11., 10.,
65          9.,  8.,  7.,  6.,  5.,  4.,  3.,  2.,  3.,  4.,  5.,
66          6.,  7.,
67          8.,  9., 10., 11., 12., 11., 10., 9., 8., 7., 6.,
68          5.,  4.,
69          3.,  2.,  3.,  4.,  5.,  6.,  7.,  8.,  9., 10., 11.])
        ]
67 [60.0 58.684500000000001
68   array([ 0.,  1.,  2.,  3.,  4.,  5.,  6.,  7.,  8.,  9., 10.,
69          11., 10.,
70          9.,  8.,  7.,  6.,  5.,  4.,  3.,  2.,  3.,  4.,  5.,
71          6.,  7.,
72          8.,  9., 10., 11., 12., 11., 10., 9., 8., 7., 6.,
73          5.,  4.,
74          3.,  2.,  3.,  4.,  5.,  6.,  7.,  8.,  9., 10., 11.])
        ]
72 [65.0 56.92485714285714
73   array([ 0.,  1.,  2.,  3.,  4.,  5.,  6.,  7.,  8.,  9., 10.,
74          11., 10.,
75          9.,  8.,  7.,  6.,  5.,  4.,  3.,  2.,  3.,  4.,  5.,
76          6.,  7.,
77          8.,  9., 10., 11., 10., 9., 8., 7., 6., 5., 4.,
78          3.,  2.,
79          3.,  4.,  5.,  6.,  7.,  8.,  9., 10., 11.])
        ]
77 [70.0 57.746175
78   array([ 0.,  1.,  2.,  3.,  4.,  5.,  6.,  7.,  8.,  9., 10.,
79          11., 10.,
          9.,  8.,  7.,  6.,  5.,  4.,  3.,  2.,  3.,  4.,  5.,
          6.,  7.,

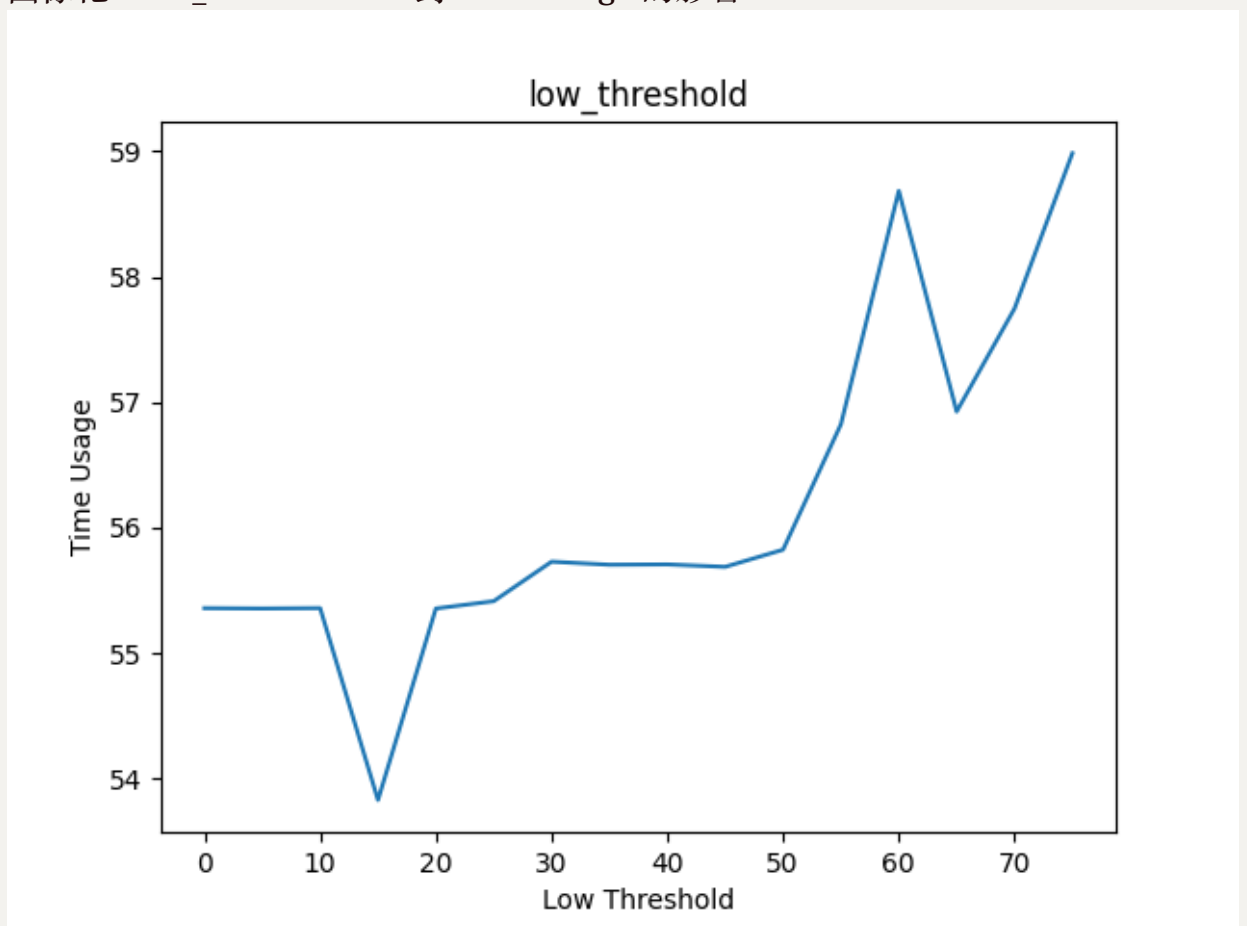
```

```

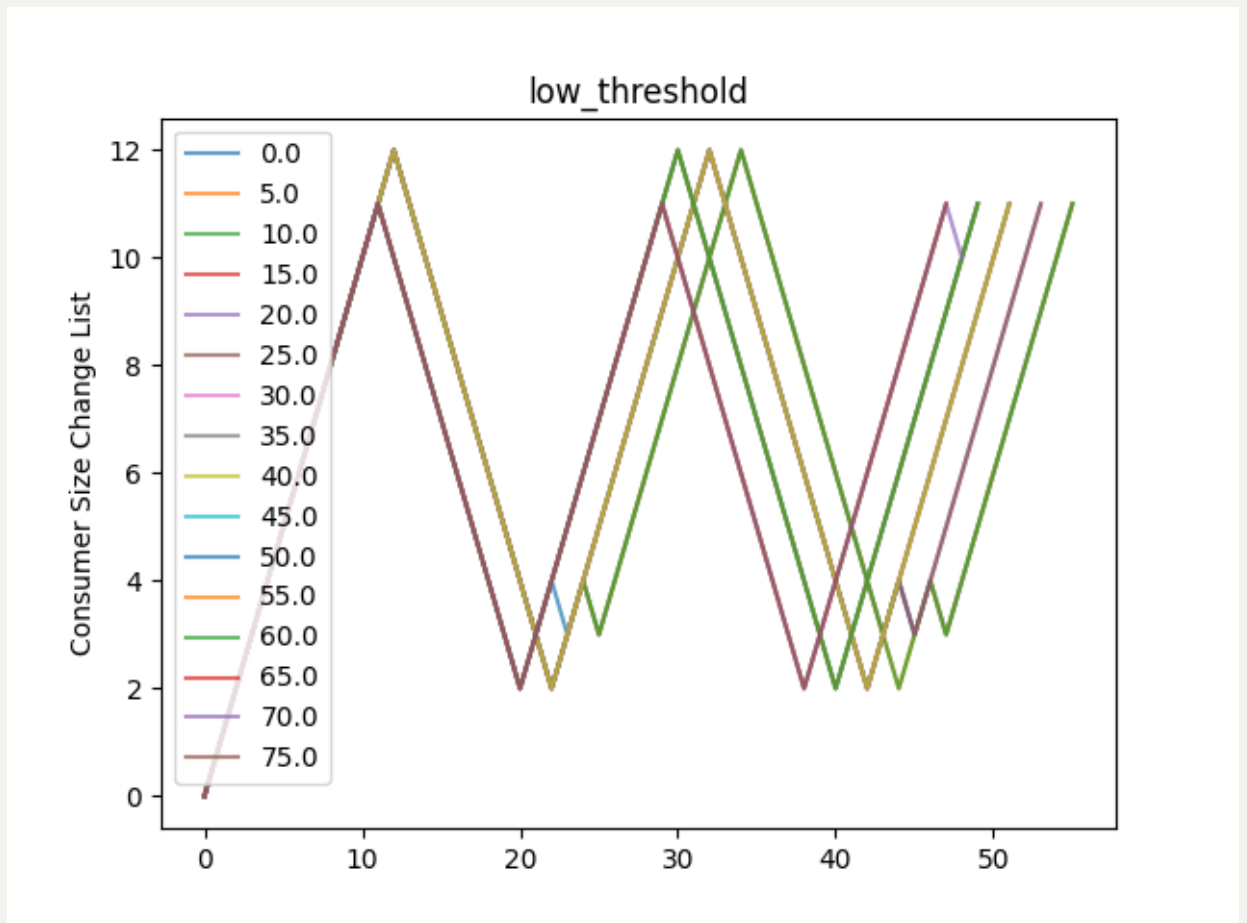
80         8., 9., 10., 11., 10., 9., 8., 7., 6., 5., 4.,
      3., 2.,
81         3., 4., 5., 6., 7., 8., 9., 10., 11., 10.]))
      ]
82 [75.0 58.9842750000000004
83  array([ 0., 1., 2., 3., 4., 5., 6., 7., 8., 9., 10.,
84  11., 10.,
85         9., 8., 7., 6., 5., 4., 3., 2., 3., 4., 5.,
86         6., 7.,
85         8., 9., 10., 11., 10., 9., 8., 7., 6., 5., 4.,
86         3., 2.,
86         3., 4., 5., 6., 7., 8., 9., 10., 11.]))
      ]]

```

1. 圖像化 LOW_THRESHOLD 對 Time Usage 的影響:



2. 圖像化 LOW_THRESHOLD 對 CONSUMER SIZE CHANGE LIST 的影響:



► CONSUMER_CONTROLLER_HIGH_THRESHOLD_PERCENTAGE 由小到大產生的 data

```

1  ['High Threshold', 'Time Usage', 'Consumer Size Change List']
2  [[25.0 55.3564125
3    array([ 0.,  1.,  2.,  3.,  4.,  5.,  6.,  7.,  8.,  9., 10.,
4    11., 12.,
5          11., 10.,  9.,  8.,  7.,  6.,  5.,  4.,  3.,  2.,  3.,
6    4.,  5.,
7          6.,  7.,  8.,  9., 10., 11., 12., 11., 10.,  9.,  8.,
8    7.,  6.,
9          5.,  4.,  3.,  2.,  3.,  4.,  3.,  4.,  5.,  6.,  7.,
10   8.,  9.,
11   10., 11.])]
12  ]
13  [30.0 55.355019999999999
14    array([ 0.,  1.,  2.,  3.,  4.,  5.,  6.,  7.,  8.,  9., 10.,
15   11., 12.,
16          11., 10.,  9.,  8.,  7.,  6.,  5.,  4.,  3.,  2.,  3.,
17   4.,  5.,
18          6.,  7.,  8.,  9., 10., 11., 12., 11., 10.,  9.,  8.,
19   7.,  6.,
20          5.,  4.,  3.,  2.,  3.,  4.,  3.,  4.,  5.,  6.,  7.,
21   8.,  9.,
22   10., 11.])]

```

```
11         6., 7., 8., 9., 10., 11., 12., 11., 10., 9., 8.,
12         7., 6.,
13         5., 4., 3., 2., 3., 4., 3., 4., 5., 6., 7.,
14         8., 9.,
15         10., 11.])
16     ]
17     [35.0 55.3563875
18     array([ 0., 1., 2., 3., 4., 5., 6., 7., 8., 9., 10.,
19            11., 12.,
20            11., 10., 9., 8., 7., 6., 5., 4., 3., 2., 3.,
21            4., 5.,
22            6., 7., 8., 9., 10., 11., 12., 11., 10., 9., 8.,
23            7., 6.,
24            5., 4., 3., 2., 3., 4., 3., 4., 5., 6., 7.,
25            8., 9.,
26            10., 11.])
27     ]
28     [40.0 55.357650000000001
29     array([ 0., 1., 2., 3., 4., 5., 6., 7., 8., 9., 10.,
30            11., 12.,
31            11., 10., 9., 8., 7., 6., 5., 4., 3., 2., 3.,
32            4., 5.,
33            6., 7., 8., 9., 10., 11., 12., 11., 10., 9., 8.,
34            7., 6.,
35            5., 4., 3., 2., 3., 4., 3., 4., 5., 6., 7.,
36            8., 9.,
37            10., 11.])
38     ]
39     [45.0 55.3583625
40     array([ 0., 1., 2., 3., 4., 5., 6., 7., 8., 9., 10.,
41            11., 12.,
42            11., 10., 9., 8., 7., 6., 5., 4., 3., 2., 3.,
43            4., 5.,
44            6., 7., 8., 9., 10., 11., 12., 11., 10., 9., 8.,
45            7., 6.,
46            5., 4., 3., 2., 3., 4., 3., 4., 5., 6., 7.,
47            8., 9.,
48            10., 11.])
49     ]
```

```
32 [50.0 55.35848571428572
33   array([ 0.,  1.,  2.,  3.,  4.,  5.,  6.,  7.,  8.,  9., 10.,
34  11., 12.,
35          11., 10.,  9.,  8.,  7.,  6.,  5.,  4.,  3.,  2.,  3.,
36  4.,  5.,
37          6.,  7.,  8.,  9., 10., 11., 12., 11., 10.,  9.,  8.,
38  7.,  6.,
39          5.,  4.,  3.,  2.,  3.,  4.,  3.,  4.,  5.,  6.,  7.,
40  8.,  9.,
41          10., 11.])
42   ]
43 [55.0 55.35292857142856
44   array([ 0.,  1.,  2.,  3.,  4.,  5.,  6.,  7.,  8.,  9., 10.,
45  11., 12.,
46          11., 10.,  9.,  8.,  7.,  6.,  5.,  4.,  3.,  2.,  3.,
47  4.,  5.,
48          6.,  7.,  8.,  9., 10., 11., 12., 11., 10.,  9.,  8.,
49  7.,  6.,
50          5.,  4.,  3.,  2.,  3.,  4.,  3.,  4.,  5.,  6.,  7.,
51  8.,  9.,
52          10., 11.])
53   ]
54 [60.0 55.353500000000004
55   array([ 0.,  1.,  2.,  3.,  4.,  5.,  6.,  7.,  8.,  9., 10.,
56  11., 12.,
57          11., 10.,  9.,  8.,  7.,  6.,  5.,  4.,  3.,  2.,  3.,
58  4.,  5.,
59          6.,  7.,  8.,  9., 10., 11., 12., 11., 10.,  9.,  8.,
60  7.,  6.,
61          5.,  4.,  3.,  2.,  3.,  4.,  3.,  4.,  5.,  6.,  7.,
62  8.,  9.,
63          10., 11.])
64   ]
65 [65.0 55.357780000000005
66   array([ 0.,  1.,  2.,  3.,  4.,  5.,  6.,  7.,  8.,  9., 10.,
67  11., 12.,
68          11., 10.,  9.,  8.,  7.,  6.,  5.,  4.,  3.,  2.,  3.,
69  4.,  5.,
70          6.,  7.,  8.,  9., 10., 11., 12., 11., 10.,  9.,  8.,
71  7.,  6.,
72          5.,  4.,  3.,  2.,  3.,  4.,  3.,  4.,  5.,  6.,  7.,
73  8.,  9.,
74          10., 11.])
75   ]
```



```

53         6., 7., 8., 9., 10., 11., 12., 11., 10., 9., 8.,
        7., 6.,
54         5., 4., 3., 2., 3., 4., 3., 4., 5., 6., 7.,
        8., 9.,
55         10., 11.])
        ]
56 [70.0 55.36524285714285
57   array([ 0., 1., 2., 3., 4., 5., 6., 7., 8., 9., 10.,
        11., 12.,
58         11., 10., 9., 8., 7., 6., 5., 4., 3., 2., 3.,
        4., 5.,
59         6., 7., 8., 9., 10., 11., 12., 11., 10., 9., 8.,
        7., 6.,
60         5., 4., 3., 2., 3., 4., 3., 4., 5., 6., 7.,
        8., 9.,
61         10., 11.])
        ]
62 [75.0 55.367016666666667
63   array([ 0., 1., 2., 3., 4., 5., 6., 7., 8., 9., 10.,
        11., 12.,
64         11., 10., 9., 8., 7., 6., 5., 4., 3., 2., 3.,
        4., 5.,
65         6., 7., 8., 9., 10., 11., 12., 11., 10., 9., 8.,
        7., 6.,
66         5., 4., 3., 2., 3., 4., 3., 4., 5., 6., 7.,
        8., 9.,
67         10., 11.])
        ]
68 [80.0 55.36472857142858
69   array([ 0., 1., 2., 3., 4., 5., 6., 7., 8., 9., 10.,
        11., 12.,
70         11., 10., 9., 8., 7., 6., 5., 4., 3., 2., 3.,
        4., 5.,
71         6., 7., 8., 9., 10., 11., 12., 11., 10., 9., 8.,
        7., 6.,
72         5., 4., 3., 2., 3., 4., 3., 4., 5., 6., 7.,
        8., 9.,
73         10., 11.])
        ]

```

```

74 [85.0 55.35898333333333
75   array([ 0.,  1.,  2.,  3.,  4.,  5.,  6.,  7.,  8.,  9., 10.,
76         11., 12.,
77         11., 10.,  9.,  8.,  7.,  6.,  5.,  4.,  3.,  2.,  3.,
78         4.,  5.,
79         6.,  7.,  8.,  9., 10., 11., 12., 11., 10.,  9.,  8.,
80         7.,  6.,
81         5.,  4.,  3.,  2.,  3.,  4.,  3.,  4.,  5.,  6.,  7.,
82         8.,  9.,
83         10., 11.])
84   ]
85 [90.0 55.35592857142858
86   array([ 0.,  1.,  2.,  3.,  4.,  5.,  6.,  7.,  8.,  9., 10.,
87         11., 12.,
88         11., 10.,  9.,  8.,  7.,  6.,  5.,  4.,  3.,  2.,  3.,
89         4.,  5.,
90         6.,  7.,  8.,  9., 10., 11., 12., 11., 10.,  9.,  8.,
91         7.,  6.,
92         5.,  4.,  3.,  2.,  3.,  4.,  3.,  4.,  5.,  6.,  7.,
93         8.,  9.,
94         10., 11.])
95   ]
96 [95.0 55.357
97   array([ 0.,  1.,  2.,  3.,  4.,  5.,  6.,  7.,  8.,  9., 10.,
98         11., 12.,
99         11., 10.,  9.,  8.,  7.,  6.,  5.,  4.,  3.,  2.,  3.,
100        4.,  5.,
101        6.,  7.,  8.,  9., 10., 11., 12., 11., 10.,  9.,  8.,
102        7.,  6.,
103        5.,  4.,  3.,  2.,  3.,  4.,  3.,  4.,  5.,  6.,  7.,
104        8.,  9.,
105        10., 11.])
106   ]
107 [100.0 55.358300000000001
108   array([ 0.,  1.,  2.,  3.,  4.,  5.,  6.,  7.,  8.,  9., 10.,
109         11., 12.,
110         11., 10.,  9.,  8.,  7.,  6.,  5.,  4.,  3.,  2.,  3.,
111         4.,  5.,

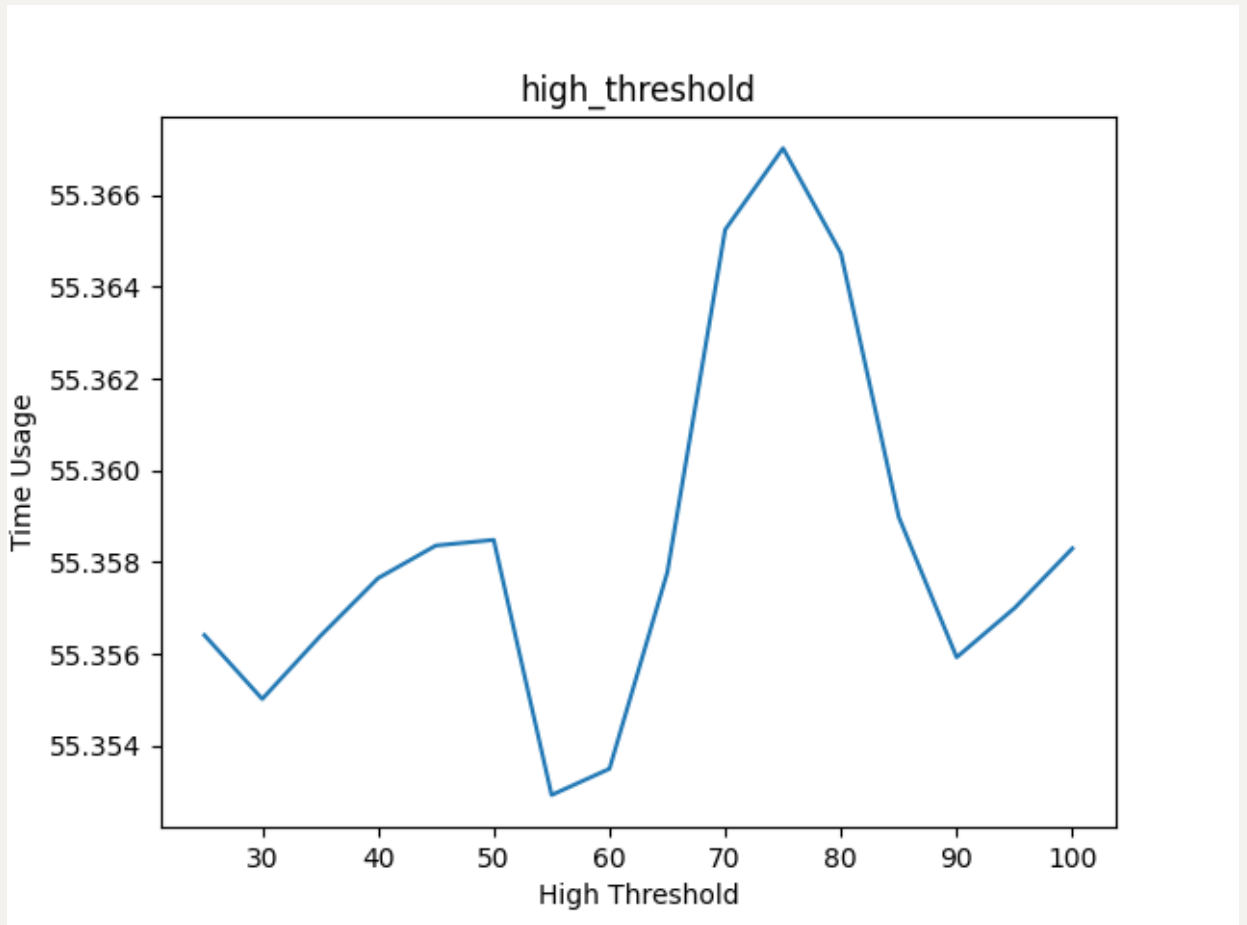
```

```

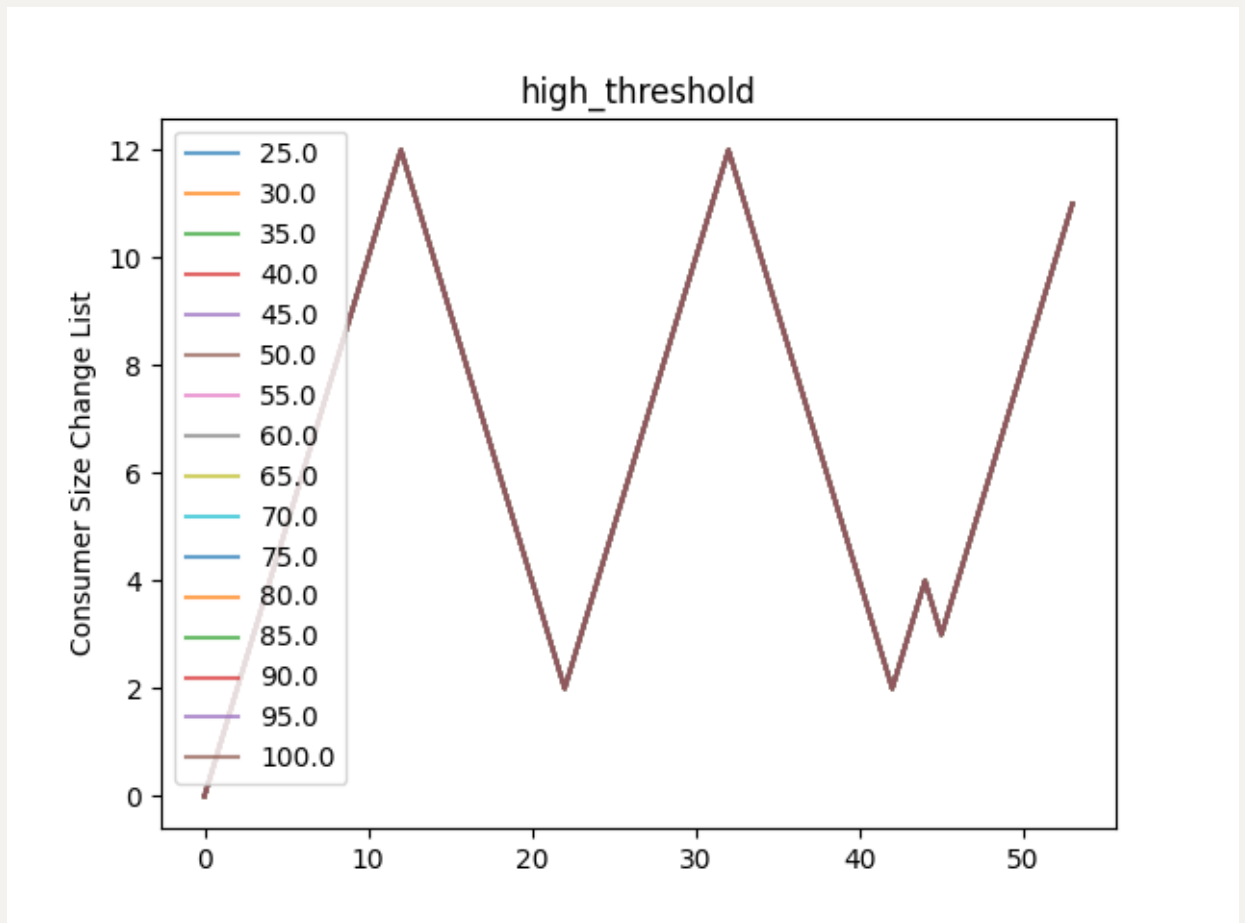
95         6., 7., 8., 9., 10., 11., 12., 11., 10., 9., 8.,
       7., 6.,
96         5., 4., 3., 2., 3., 4., 3., 4., 5., 6., 7.,
       8., 9.,
97         10., 11.])
       11

```

1. 圖像化 HIGH_THRESHOLD 對 Time Usage 的影響



2. 圖像化 HIGH_THRESHOLD 對 CONSUMER SIZE CHANGE LIST 的影響:



3. 討論

- a. 觀看兩者變因的圖, 我們可以發現 Time Usage 在 LOW THRESHOLD 有最低值, 約在 Threshold = 15 時. 也可以大致觀察出, 在 low threshold Time Usage 較小, 但反觀 HIGH_THRESHOLD 的圖, 似乎 HIGH THRESHOLD 值的改變對 Time Usage 影響不大
- b. HIGH THRESHOLD 的變化對 CONSUMER SIZE CHANGE 沒有影響, 所有線是疊在一起的, 呈現波谷波峰波谷波峰, 而 LOW THRESHOLD 的圖則可以看出 LOW THRESHOLD 低的, 可以達到最高峰 (震幅較大), 改變次數也減少(可以對應到上一點, LOW THRESHOLD Time Usage 較小), 總體波形成現谷峰谷峰谷峰 (看圖可能因為重疊而難以分辨, 可以 txt 文字 data 輔助)

Different values of WORKER_QUEUE_SIZE

► WORKER_QUEUE_SIZE 由小到大所產生的 data

```
1  ['Worker Queue Size', 'Time Usage', 'Consumer Size Change List']
2  [[20.0 59.941537499999995
3     array([ 0.,  1.,  2.,  3.,  4.,  5.,  6.,  7.,  8.,  9., 10.,
            9.,  8.,
```

```

4         7., 6., 5., 4., 3., 2., 3., 4., 3., 2., 3.,
      4., 3.,
5         2., 3., 4., 5., 6., 7., 8., 9., 10., 9., 8.,
      7., 6.,
6         5., 4., 3., 2., 3., 4., 3., 2., 3., 4., 3.,
      2., 3.,
7         4., 5., 6., 7., 8., 9., 10., 9.])
      ]
8 [40.0 57.495849999999999
9  array([ 0., 1., 2., 3., 4., 5., 6., 7., 8., 9., 10.,
10 11., 10.,
11         9., 8., 7., 6., 5., 4., 3., 2., 3., 4., 3.,
12 2., 3.,
13         4., 5., 6., 7., 8., 9., 10., 11., 10., 9., 8.,
14 7., 6.,
15         5., 4., 3., 2., 3., 4., 3., 2., 3., 4., 3.,
16 4., 5.,
17         6., 7., 8., 9., 10., 11.])
      ]
14 [60.0 57.333587499999999
15  array([ 0. , 1. , 2. , 3. , 4. , 5. , 6. , 7. ,
16 8. ,
17         9. , 10. , 11. , 10. , 9. , 8. , 7. , 6. ,
18 5. ,
19         4. , 3. , 2. , 3. , 4. , 3. , 2. , 3. ,
20 4. ,
21         5. , 6. , 7. , 8. , 9. , 10. , 11. , 10.25,
22 9.25,
23         8.25, 7.25, 6.25, 5.25, 4.25, 3.25, 2.25, 3. ,
24 3.75,
25         3. , 2.25, 3. , 3.75, 3. , 4. , 5. , 6. ,
26 7. ,
27         8. , 9. , 10. , 11.  ] )
      ]
22 [80.0 59.272187500000001
23  array([ 0., 1., 2., 3., 4., 5., 6., 7., 8., 9., 10.,
24 11., 12.,
25         11., 10., 9., 8., 7., 6., 5., 4., 3., 2., 3.,
26 4., 3.,

```

```

25         2., 3., 4., 5., 6., 7., 8., 9., 10., 11., 12.,
11., 10.,
26         9., 8., 7., 6., 5., 4., 3., 2., 3., 4., 3.,
2., 3.,
27         4., 5., 6., 7., 8., 9., 10., 11.]))
    ]
28 [100.0 59.217512500000005
29   array([ 0., 1., 2., 3., 4., 5., 6., 7., 8., 9., 10.,
11., 12.,
30         11., 10., 9., 8., 7., 6., 5., 4., 3., 2., 3.,
4., 3.,
31         2., 3., 4., 5., 6., 7., 8., 9., 10., 11., 12.,
11., 10.,
32         9., 8., 7., 6., 5., 4., 3., 2., 3., 4., 3.,
2., 3.,
33         4., 5., 6., 7., 8., 9., 10., 11.]))
    ]
34 [120.0 59.2023
35   array([ 0., 1., 2., 3., 4., 5., 6., 7., 8., 9., 10.,
11., 12.,
36         11., 10., 9., 8., 7., 6., 5., 4., 3., 2., 3.,
4., 3.,
37         2., 3., 4., 5., 6., 7., 8., 9., 10., 11., 12.,
11., 10.,
38         9., 8., 7., 6., 5., 4., 3., 2., 3., 4., 3.,
2., 3.,
39         4., 5., 6., 7., 8., 9., 10., 11.]))
    ]
40 [140.0 54.84546
41   array([ 0., 1., 2., 3., 4., 5., 6., 7., 8., 9., 10.,
11., 12.,
42         11., 10., 9., 8., 7., 6., 5., 4., 3., 2., 3.,
4., 3.,
43         4., 5., 6., 7., 8., 9., 10., 11., 12., 11., 10.,
9., 8.,
44         7., 6., 5., 4., 3., 2., 3., 4., 5., 6., 7.,
8., 9.,
45         10., 11.]))
    ]

```

```

46 [160.0 55.45547142857142
47   array([ 0.,  1.,  2.,  3.,  4.,  5.,  6.,  7.,  8.,  9., 10.,
48         11., 12.,
49           11., 10.,  9.,  8.,  7.,  6.,  5.,  4.,  3.,  2.,  3.,
50           4.,  5.,
51           6.,  7.,  8.,  9., 10., 11., 12., 11., 10.,  9.,  8.,
52           7.,  6.,
53           5.,  4.,  3.,  2.,  3.,  4.,  3.,  4.,  5.,  6.,  7.,
54           8.,  9.,
55           10., 11.])
56   ]
57 [180.0 55.410475
58   array([ 0.,  1.,  2.,  3.,  4.,  5.,  6.,  7.,  8.,  9., 10.,
59         11., 12.,
60           11., 10.,  9.,  8.,  7.,  6.,  5.,  4.,  3.,  2.,  3.,
61           4.,  5.,
62           6.,  7.,  8.,  9., 10., 11., 12., 11., 10.,  9.,  8.,
63           7.,  6.,
64           5.,  4.,  3.,  2.,  3.,  4.,  3.,  4.,  5.,  6.,  7.,
65           8.,  9.,
66           10., 11.])
67   ]
68 [200.0 55.353366666666666
69   array([ 0.,  1.,  2.,  3.,  4.,  5.,  6.,  7.,  8.,  9., 10.,
70         11., 12.,
71           11., 10.,  9.,  8.,  7.,  6.,  5.,  4.,  3.,  2.,  3.,
72           4.,  5.,
73           6.,  7.,  8.,  9., 10., 11., 12., 11., 10.,  9.,  8.,
74           7.,  6.,
75           5.,  4.,  3.,  2.,  3.,  4.,  3.,  4.,  5.,  6.,  7.,
76           8.,  9.,
77           10., 11.])
78   ]
79 [220.0 53.776812500000005
80   array([ 0.,  1.,  2.,  3.,  4.,  5.,  6.,  7.,  8.,  9., 10.,
81         11., 12.,
82           11., 10.,  9.,  8.,  7.,  6.,  5.,  4.,  3.,  2.,  3.,
83           4.,  5.,

```

```

67         6., 7., 8., 9., 10., 11., 12., 11., 10., 9., 8.,
        7., 6.,
68         5., 4., 3., 2., 3., 4., 5., 6., 7., 8., 9.,
10., 11.]]]
69 [240.0 53.753575
70  array([ 0., 1., 2., 3., 4., 5., 6., 7., 8., 9., 10.,
11., 12.,
71         11., 10., 9., 8., 7., 6., 5., 4., 3., 2., 3.,
        4., 5.,
72         6., 7., 8., 9., 10., 11., 12., 11., 10., 9., 8.,
        7., 6.,
73         5., 4., 3., 2., 3., 4., 5., 6., 7., 8., 9.,
10., 11.]]]
74 [260.0 53.7031875
75  array([ 0., 1., 2., 3., 4., 5., 6., 7., 8., 9., 10.,
11., 12.,
76         11., 10., 9., 8., 7., 6., 5., 4., 3., 2., 3.,
        4., 5.,
77         6., 7., 8., 9., 10., 11., 12., 11., 10., 9., 8.,
        7., 6.,
78         5., 4., 3., 2., 3., 4., 5., 6., 7., 8., 9.,
10., 11.]]]
79 [280.0 54.0351875
80  array([ 0., 1., 2., 3., 4., 5., 6., 7., 8., 9., 10.,
11., 12.,
81         11., 10., 9., 8., 7., 6., 5., 4., 3., 2., 3.,
        4., 5.,
82         6., 7., 8., 9., 10., 11., 12., 11., 10., 9., 8.,
        7., 6.,
83         5., 4., 3., 2., 3., 4., 5., 6., 7., 8., 9.,
10., 11.]]]
84 [300.0 55.501787500000006
85  array([ 0., 1., 2., 3., 4., 5., 6., 7., 8., 9., 10.,
11., 12.,
86         11., 10., 9., 8., 7., 6., 5., 4., 3., 2., 3.,
        4., 5.,
87         6., 7., 8., 9., 10., 11., 12., 11., 10., 9., 8.,
        7., 6.,

```



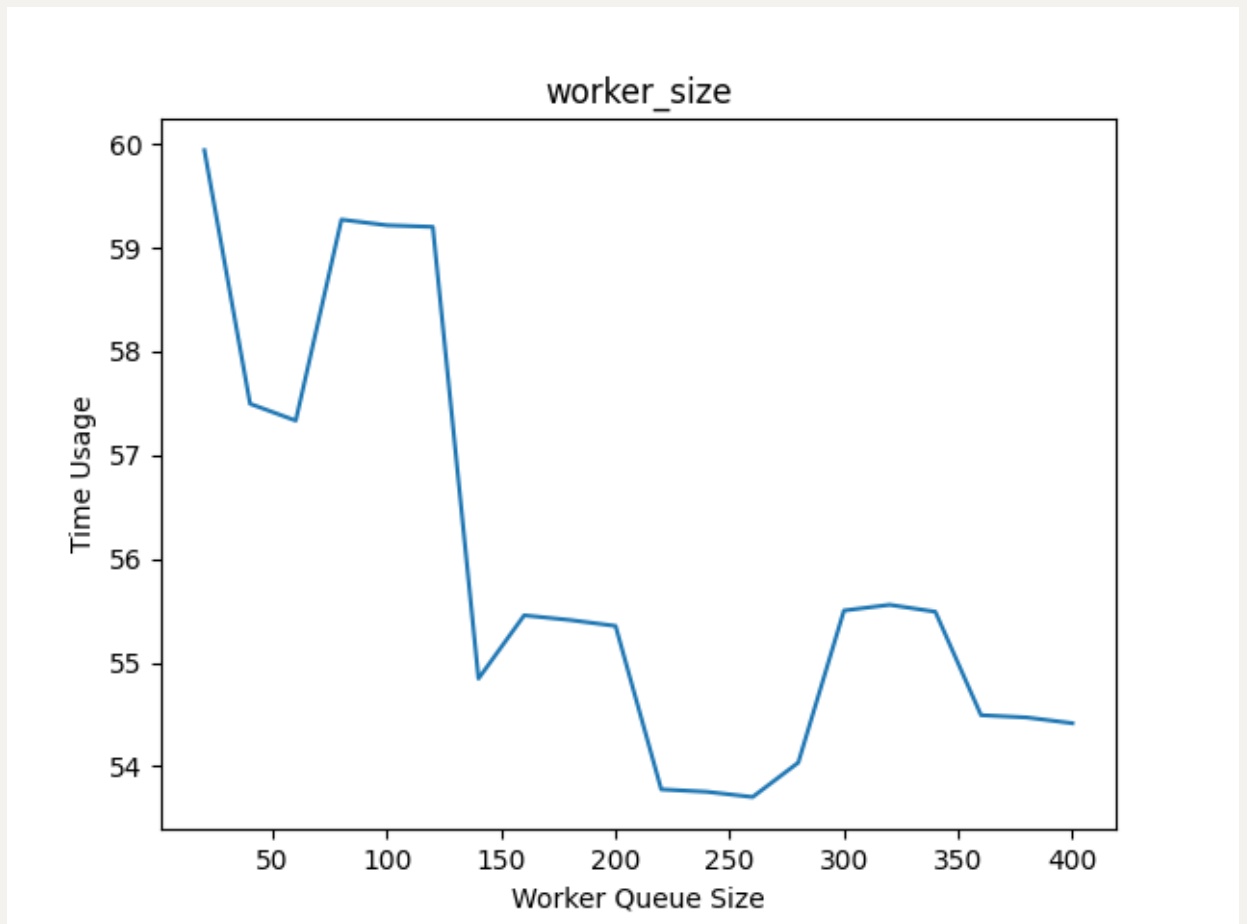
```
88         5., 4., 3., 2., 3., 4., 5., 6., 7., 8., 9.,
      10., 11.]]]
89 [320.0 55.5569625000000004
90   array([ 0., 1., 2., 3., 4., 5., 6., 7., 8., 9., 10.,
      11., 12.,
91         11., 10., 9., 8., 7., 6., 5., 4., 3., 2., 3.,
      4., 5.,
92         6., 7., 8., 9., 10., 11., 12., 11., 10., 9., 8.,
      7., 6.,
93         5., 4., 3., 2., 3., 4., 5., 6., 7., 8., 9.,
      10., 11.]]])
94 [340.0 55.4902
95   array([ 0., 1., 2., 3., 4., 5., 6., 7., 8., 9., 10.,
      11., 12.,
96         11., 10., 9., 8., 7., 6., 5., 4., 3., 2., 3.,
      4., 5.,
97         6., 7., 8., 9., 10., 11., 12., 11., 10., 9., 8.,
      7., 6.,
98         5., 4., 3., 2., 3., 4., 5., 6., 7., 8., 9.,
      10., 11.]]])
99 [360.0 54.492525
100   array([ 0., 1., 2., 3., 4., 5., 6., 7., 8., 9., 10.,
      11., 12.,
101         11., 10., 9., 8., 7., 6., 5., 4., 3., 2., 3.,
      4., 5.,
102         6., 7., 8., 9., 10., 11., 12., 11., 10., 9., 8.,
      7., 6.,
103         5., 4., 3., 2., 3., 4., 5., 6., 7., 8., 9.,
      10., 11.]]])
104 [380.0 54.4706125
105   array([ 0., 1., 2., 3., 4., 5., 6., 7., 8., 9., 10.,
      11., 12.,
106         11., 10., 9., 8., 7., 6., 5., 4., 3., 2., 3.,
      4., 5.,
107         6., 7., 8., 9., 10., 11., 12., 11., 10., 9., 8.,
      7., 6.,
108         5., 4., 3., 2., 3., 4., 5., 6., 7., 8., 9.,
      10., 11.]]])
109 [400.0 54.414837499999999]
```

```

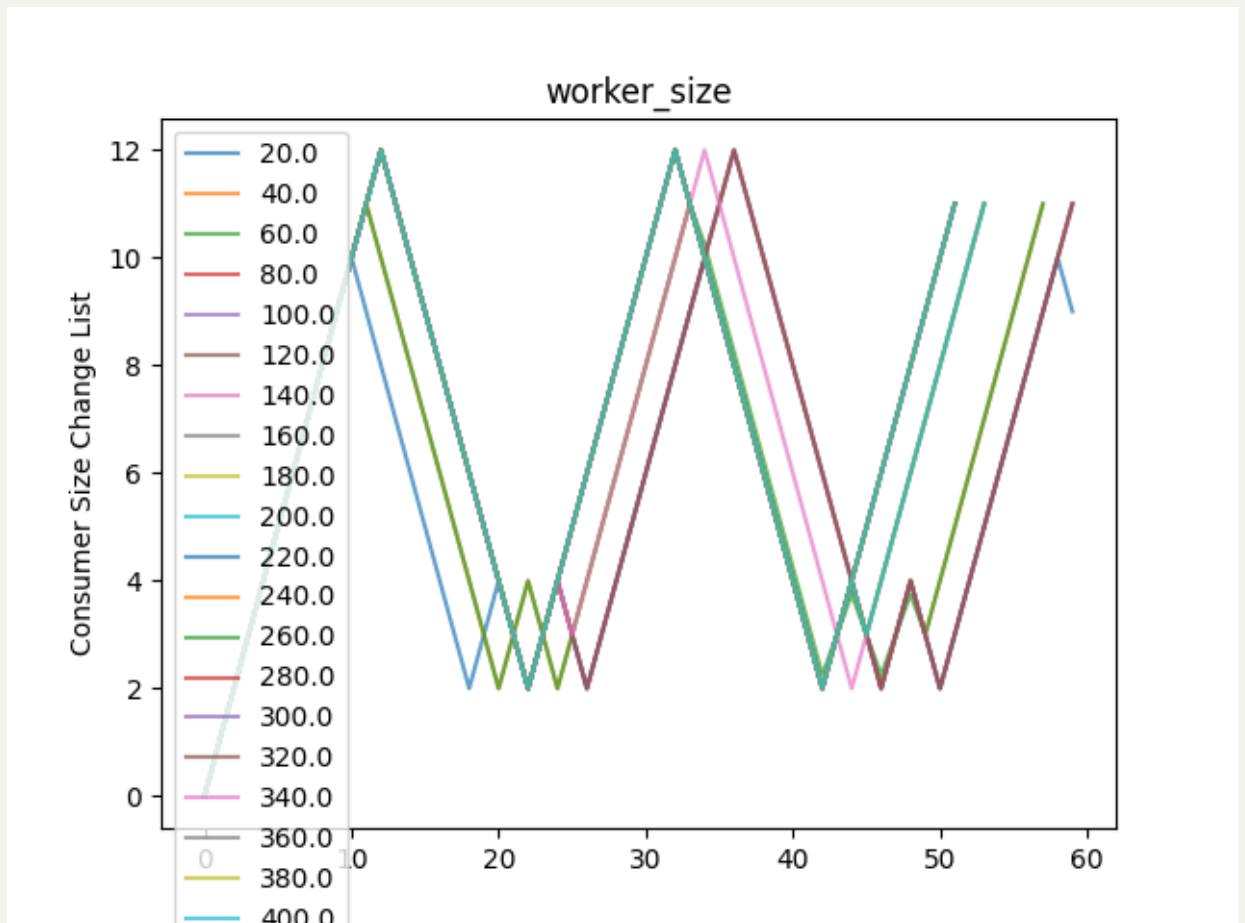
110     array([ 0.,  1.,  2.,  3.,  4.,  5.,  6.,  7.,  8.,  9., 10.,
111            11., 12.,
112            11., 10.,  9.,  8.,  7.,  6.,  5.,  4.,  3.,  2.,  3.,
113            4.,  5.,
114            6.,  7.,  8.,  9., 10., 11., 12., 11., 10.,  9.,  8.,
115            7.,  6.,
116            5.,  4.,  3.,  2.,  3.,  4.,  5.,  6.,  7.,  8.,  9.,
117            10., 11.]])

```

1. 圖像化 WORKER_QUEUE_SIZE 對 Time Usage 的影響:



2. 圖像化 WRITER_QUEUE_SIZE 對 CONSUMER SIZE CHANGE LIST 的影響:



3. 討論:

- 可以發現 WORKER_QUEUE_SIZE 和 Time Usage 有點呈現階梯狀下降, WORKER_QUEUE_SIZE 越大, Time Usage 越小
- WORKER_QUEUE_SIZE 越大可以達到 COMSUMER SIZE 的峰值 12, 較小的還未達到 12 即下降 (震幅較小), 總體波形呈現谷峰谷峰谷峰

What happens if WRITER_QUEUE_SIZE is very small?

► WRITER_QUEUE_SIZE 由小到大所產生的 data:

```
1 ['Writer Queue Size', 'Time Usage', 'Consumer Size Change List']
2 [[250.0 55.3548375
3    array([ 0.,  1.,  2.,  3.,  4.,  5.,  6.,  7.,  8.,  9., 10.,
4          11., 12.,
5            11., 10.,  9.,  8.,  7.,  6.,  5.,  4.,  3.,  2.,  3.,
6            4.,  5.,
7            6.,  7.,  8.,  9., 10., 11., 12., 11., 10.,  9.,  8.,
8            7.,  6.,
9            5.,  4.,  3.,  2.,  3.,  4.,  3.,  4.,  5.,  6.,  7.,
10           8.,  9.,
```

```

7         10., 11.])
8     ]
9     [500.0 55.3569
10    array([ 0.,  1.,  2.,  3.,  4.,  5.,  6.,  7.,  8.,  9., 10.,
11           11., 12.,
12           11., 10.,  9.,  8.,  7.,  6.,  5.,  4.,  3.,  2.,  3.,
13           4.,  5.,
14           6.,  7.,  8.,  9., 10., 11., 12., 11., 10.,  9.,  8.,
15           7.,  6.,
16           5.,  4.,  3.,  2.,  3.,  4.,  3.,  4.,  5.,  6.,  7.,
17           8.,  9.,
18           10., 11.])
19    ]
20    [750.0 55.3546875
21    array([ 0.,  1.,  2.,  3.,  4.,  5.,  6.,  7.,  8.,  9., 10.,
22           11., 12.,
23           11., 10.,  9.,  8.,  7.,  6.,  5.,  4.,  3.,  2.,  3.,
24           4.,  5.,
25           6.,  7.,  8.,  9., 10., 11., 12., 11., 10.,  9.,  8.,
26           7.,  6.,
27           5.,  4.,  3.,  2.,  3.,  4.,  3.,  4.,  5.,  6.,  7.,
28           8.,  9.,
29           10., 11.])
30    ]
31    [1000.0 55.353199999999994
32    array([ 0.,  1.,  2.,  3.,  4.,  5.,  6.,  7.,  8.,  9., 10.,
33           11., 12.,
34           11., 10.,  9.,  8.,  7.,  6.,  5.,  4.,  3.,  2.,  3.,
35           4.,  5.,
36           6.,  7.,  8.,  9., 10., 11., 12., 11., 10.,  9.,  8.,
37           7.,  6.,
38           5.,  4.,  3.,  2.,  3.,  4.,  3.,  4.,  5.,  6.,  7.,
39           8.,  9.,
40           10., 11.])
41    ]
42    [1250.0 55.354949999999995
43    array([ 0.,  1.,  2.,  3.,  4.,  5.,  6.,  7.,  8.,  9., 10.,
44           11., 12.,

```

```

28         11., 10., 9., 8., 7., 6., 5., 4., 3., 2., 3.,
    4., 5.,
29         6., 7., 8., 9., 10., 11., 12., 11., 10., 9., 8.,
    7., 6.,
30         5., 4., 3., 2., 3., 4., 3., 4., 5., 6., 7.,
    8., 9.,
31         10., 11.])
    ]
32 [1500.0 55.356949999999999
33  array([ 0.,  1.,  2.,  3.,  4.,  5.,  6.,  7.,  8.,  9., 10.,
    11., 12.,
34         11., 10., 9., 8., 7., 6., 5., 4., 3., 2., 3.,
    4., 5.,
35         6., 7., 8., 9., 10., 11., 12., 11., 10., 9., 8.,
    7., 6.,
36         5., 4., 3., 2., 3., 4., 3., 4., 5., 6., 7.,
    8., 9.,
37         10., 11.])
    ]
38 [1750.0 55.357949999999999
39  array([ 0.,  1.,  2.,  3.,  4.,  5.,  6.,  7.,  8.,  9., 10.,
    11., 12.,
40         11., 10., 9., 8., 7., 6., 5., 4., 3., 2., 3.,
    4., 5.,
41         6., 7., 8., 9., 10., 11., 12., 11., 10., 9., 8.,
    7., 6.,
42         5., 4., 3., 2., 3., 4., 3., 4., 5., 6., 7.,
    8., 9.,
43         10., 11.])
    ]
44 [2000.0 55.356775
45  array([ 0.,  1.,  2.,  3.,  4.,  5.,  6.,  7.,  8.,  9., 10.,
    11., 12.,
46         11., 10., 9., 8., 7., 6., 5., 4., 3., 2., 3.,
    4., 5.,
47         6., 7., 8., 9., 10., 11., 12., 11., 10., 9., 8.,
    7., 6.,
48         5., 4., 3., 2., 3., 4., 3., 4., 5., 6., 7.,
    8., 9.,

```

```

49         10., 11.])
50     ]
51     [2250.0 55.356
52     array([ 0.,  1.,  2.,  3.,  4.,  5.,  6.,  7.,  8.,  9., 10.,
53            11., 12.,
54            11., 10.,  9.,  8.,  7.,  6.,  5.,  4.,  3.,  2.,  3.,
55            4.,  5.,
56            6.,  7.,  8.,  9., 10., 11., 12., 11., 10.,  9.,  8.,
57            7.,  6.,
58            5.,  4.,  3.,  2.,  3.,  4.,  3.,  4.,  5.,  6.,  7.,
59            8.,  9.,
60            10., 11.])
61     ]
62     [2500.0 55.3562800000000005
63     array([ 0.,  1.,  2.,  3.,  4.,  5.,  6.,  7.,  8.,  9., 10.,
64            11., 12.,
65            11., 10.,  9.,  8.,  7.,  6.,  5.,  4.,  3.,  2.,  3.,
66            4.,  5.,
67            6.,  7.,  8.,  9., 10., 11., 12., 11., 10.,  9.,  8.,
68            7.,  6.,
69            5.,  4.,  3.,  2.,  3.,  4.,  3.,  4.,  5.,  6.,  7.,
70            8.,  9.,
71            10., 11.])
72     ]
73     [2750.0 55.35272
74     array([ 0.,  1.,  2.,  3.,  4.,  5.,  6.,  7.,  8.,  9., 10.,
75            11., 12.,
76            11., 10.,  9.,  8.,  7.,  6.,  5.,  4.,  3.,  2.,  3.,
77            4.,  5.,
78            6.,  7.,  8.,  9., 10., 11., 12., 11., 10.,  9.,  8.,
79            7.,  6.,
80            5.,  4.,  3.,  2.,  3.,  4.,  3.,  4.,  5.,  6.,  7.,
81            8.,  9.,
82            10., 11.])
83     ]
84     [3000.0 55.355533333333334
85     array([ 0.,  1.,  2.,  3.,  4.,  5.,  6.,  7.,  8.,  9., 10.,
86            11., 12.,

```

```
70         11., 10., 9., 8., 7., 6., 5., 4., 3., 2., 3.,
4., 5.,
71         6., 7., 8., 9., 10., 11., 12., 11., 10., 9., 8.,
7., 6.,
72         5., 4., 3., 2., 3., 4., 3., 4., 5., 6., 7.,
8., 9.,
73         10., 11.])
    ]
74 [3250.0 55.357450000000001
75  array([ 0., 1., 2., 3., 4., 5., 6., 7., 8., 9., 10.,
11., 12.,
76         11., 10., 9., 8., 7., 6., 5., 4., 3., 2., 3.,
4., 5.,
77         6., 7., 8., 9., 10., 11., 12., 11., 10., 9., 8.,
7., 6.,
78         5., 4., 3., 2., 3., 4., 3., 4., 5., 6., 7.,
8., 9.,
79         10., 11.])
    ]
80 [3500.0 55.35671428571429
81  array([ 0., 1., 2., 3., 4., 5., 6., 7., 8., 9., 10.,
11., 12.,
82         11., 10., 9., 8., 7., 6., 5., 4., 3., 2., 3.,
4., 5.,
83         6., 7., 8., 9., 10., 11., 12., 11., 10., 9., 8.,
7., 6.,
84         5., 4., 3., 2., 3., 4., 3., 4., 5., 6., 7.,
8., 9.,
85         10., 11.])
    ]
86 [3750.0 55.3678125
87  array([ 0., 1., 2., 3., 4., 5., 6., 7., 8., 9., 10.,
11., 12.,
88         11., 10., 9., 8., 7., 6., 5., 4., 3., 2., 3.,
4., 5.,
89         6., 7., 8., 9., 10., 11., 12., 11., 10., 9., 8.,
7., 6.,
90         5., 4., 3., 2., 3., 4., 3., 4., 5., 6., 7.,
8., 9.,
```

```

91         10., 11.])
92     ]
93     [4000.0 55.358075
94     array([ 0.,  1.,  2.,  3.,  4.,  5.,  6.,  7.,  8.,  9., 10.,
11., 12.,
95           11., 10.,  9.,  8.,  7.,  6.,  5.,  4.,  3.,  2.,  3.,
96           4.,  5.,
97           6.,  7.,  8.,  9., 10., 11., 12., 11., 10.,  9.,  8.,
98           7.,  6.,
99           5.,  4.,  3.,  2.,  3.,  4.,  3.,  4.,  5.,  6.,  7.,
100          8.,  9.,
101          10., 11.])
102     ]
103     [4250.0 55.3819125
104     array([ 0.,  1.,  2.,  3.,  4.,  5.,  6.,  7.,  8.,  9., 10.,
105          11., 12.,
106          11., 10.,  9.,  8.,  7.,  6.,  5.,  4.,  3.,  2.,  3.,
107          4.,  5.,
108          6.,  7.,  8.,  9., 10., 11., 12., 11., 10.,  9.,  8.,
109          7.,  6.,
110          5.,  4.,  3.,  2.,  3.,  4.,  3.,  4.,  5.,  6.,  7.,
111          8.,  9.,
112          10., 11.])
113     ]
114     [4500.0 55.359871428571424
115     array([ 0.,  1.,  2.,  3.,  4.,  5.,  6.,  7.,  8.,  9., 10.,
116          11., 12.,
117          11., 10.,  9.,  8.,  7.,  6.,  5.,  4.,  3.,  2.,  3.,
118          4.,  5.,
119          6.,  7.,  8.,  9., 10., 11., 12., 11., 10.,  9.,  8.,
120          7.,  6.,
121          5.,  4.,  3.,  2.,  3.,  4.,  3.,  4.,  5.,  6.,  7.,
122          8.,  9.,
123          10., 11.])
124     ]
125     [4750.0 55.35838333333333
126     array([ 0.,  1.,  2.,  3.,  4.,  5.,  6.,  7.,  8.,  9., 10.,
127          11., 12.,

```



```

112         11., 10., 9., 8., 7., 6., 5., 4., 3., 2., 3.,
        4., 5.,
113         6., 7., 8., 9., 10., 11., 12., 11., 10., 9., 8.,
        7., 6.,
114         5., 4., 3., 2., 3., 4., 3., 4., 5., 6., 7.,
        8., 9.,
115         10., 11.])
        ]
116 [5000.0 55.360549999999996
117  array([ 0.,  1.,  2.,  3.,  4.,  5.,  6.,  7.,  8.,  9., 10.,
118         11., 12.,
119         11., 10., 9., 8., 7., 6., 5., 4., 3., 2., 3.,
        4., 5.,
120         6., 7., 8., 9., 10., 11., 12., 11., 10., 9., 8.,
        7., 6.,
121         5., 4., 3., 2., 3., 4., 3., 4., 5., 6., 7.,
        8., 9.,
122         10., 11.])
        ]
123 [5250.0 55.35682857142858
124  array([ 0.,  1.,  2.,  3.,  4.,  5.,  6.,  7.,  8.,  9., 10.,
125         11., 12.,
126         11., 10., 9., 8., 7., 6., 5., 4., 3., 2., 3.,
        4., 5.,
127         6., 7., 8., 9., 10., 11., 12., 11., 10., 9., 8.,
        7., 6.,
128         5., 4., 3., 2., 3., 4., 3., 4., 5., 6., 7.,
        8., 9.,
129         10., 11.])
        ]
130 [5500.0 55.3578
131  array([ 0.,  1.,  2.,  3.,  4.,  5.,  6.,  7.,  8.,  9., 10.,
132         11., 12.,
        11., 10., 9., 8., 7., 6., 5., 4., 3., 2., 3.,
        4., 5.,
        6., 7., 8., 9., 10., 11., 12., 11., 10., 9., 8.,
        7., 6.,
        5., 4., 3., 2., 3., 4., 3., 4., 5., 6., 7.,
        8., 9.,

```

```

133         10., 11.])
134     ]
135     [5750.0 55.363075
136     array([ 0.,  1.,  2.,  3.,  4.,  5.,  6.,  7.,  8.,  9., 10.,
137            11., 12.,
138            11., 10.,  9.,  8.,  7.,  6.,  5.,  4.,  3.,  2.,  3.,
139            4.,  5.,
140            6.,  7.,  8.,  9., 10., 11., 12., 11., 10.,  9.,  8.,
141            7.,  6.,
142            5.,  4.,  3.,  2.,  3.,  4.,  3.,  4.,  5.,  6.,  7.,
143            8.,  9.,
144            10., 11.])
145     ]
146     [6000.0 55.36355714285715
147     array([ 0.,  1.,  2.,  3.,  4.,  5.,  6.,  7.,  8.,  9., 10.,
148            11., 12.,
149            11., 10.,  9.,  8.,  7.,  6.,  5.,  4.,  3.,  2.,  3.,
150            4.,  5.,
151            6.,  7.,  8.,  9., 10., 11., 12., 11., 10.,  9.,  8.,
152            7.,  6.,
153            5.,  4.,  3.,  2.,  3.,  4.,  3.,  4.,  5.,  6.,  7.,
154            8.,  9.,
155            10., 11.])
156     ]
157     [6250.0 55.357342857142854
158     array([ 0.,  1.,  2.,  3.,  4.,  5.,  6.,  7.,  8.,  9., 10.,
159            11., 12.,
160            11., 10.,  9.,  8.,  7.,  6.,  5.,  4.,  3.,  2.,  3.,
161            4.,  5.,
162            6.,  7.,  8.,  9., 10., 11., 12., 11., 10.,  9.,  8.,
163            7.,  6.,
164            5.,  4.,  3.,  2.,  3.,  4.,  3.,  4.,  5.,  6.,  7.,
165            8.,  9.,
166            10., 11.])
167     ]
168     [6500.0 55.35555
169     array([ 0.,  1.,  2.,  3.,  4.,  5.,  6.,  7.,  8.,  9., 10.,
170            11., 12.,

```

```

154         11., 10., 9., 8., 7., 6., 5., 4., 3., 2., 3.,
        4., 5.,
155         6., 7., 8., 9., 10., 11., 12., 11., 10., 9., 8.,
        7., 6.,
156         5., 4., 3., 2., 3., 4., 3., 4., 5., 6., 7.,
        8., 9.,
157         10., 11.])
        ]
158 [6750.0 55.3618
159 array([ 0., 1., 2., 3., 4., 5., 6., 7., 8., 9., 10.,
160 11., 12.,
161         11., 10., 9., 8., 7., 6., 5., 4., 3., 2., 3.,
        4., 5.,
162         6., 7., 8., 9., 10., 11., 12., 11., 10., 9., 8.,
        7., 6.,
163         5., 4., 3., 2., 3., 4., 3., 4., 5., 6., 7.,
        8., 9.,
164         10., 11.])
        ]
164 [7000.0 55.355183333333334
165 array([ 0., 1., 2., 3., 4., 5., 6., 7., 8., 9., 10.,
166 11., 12.,
167         11., 10., 9., 8., 7., 6., 5., 4., 3., 2., 3.,
        4., 5.,
168         6., 7., 8., 9., 10., 11., 12., 11., 10., 9., 8.,
        7., 6.,
169         5., 4., 3., 2., 3., 4., 3., 4., 5., 6., 7.,
        8., 9.,
170         10., 11.])
        ]
170 [7250.0 55.35501666666666664
171 array([ 0., 1., 2., 3., 4., 5., 6., 7., 8., 9., 10.,
172 11., 12.,
173         11., 10., 9., 8., 7., 6., 5., 4., 3., 2., 3.,
        4., 5.,
174         6., 7., 8., 9., 10., 11., 12., 11., 10., 9., 8.,
        7., 6.,
        5., 4., 3., 2., 3., 4., 3., 4., 5., 6., 7.,
        8., 9.,

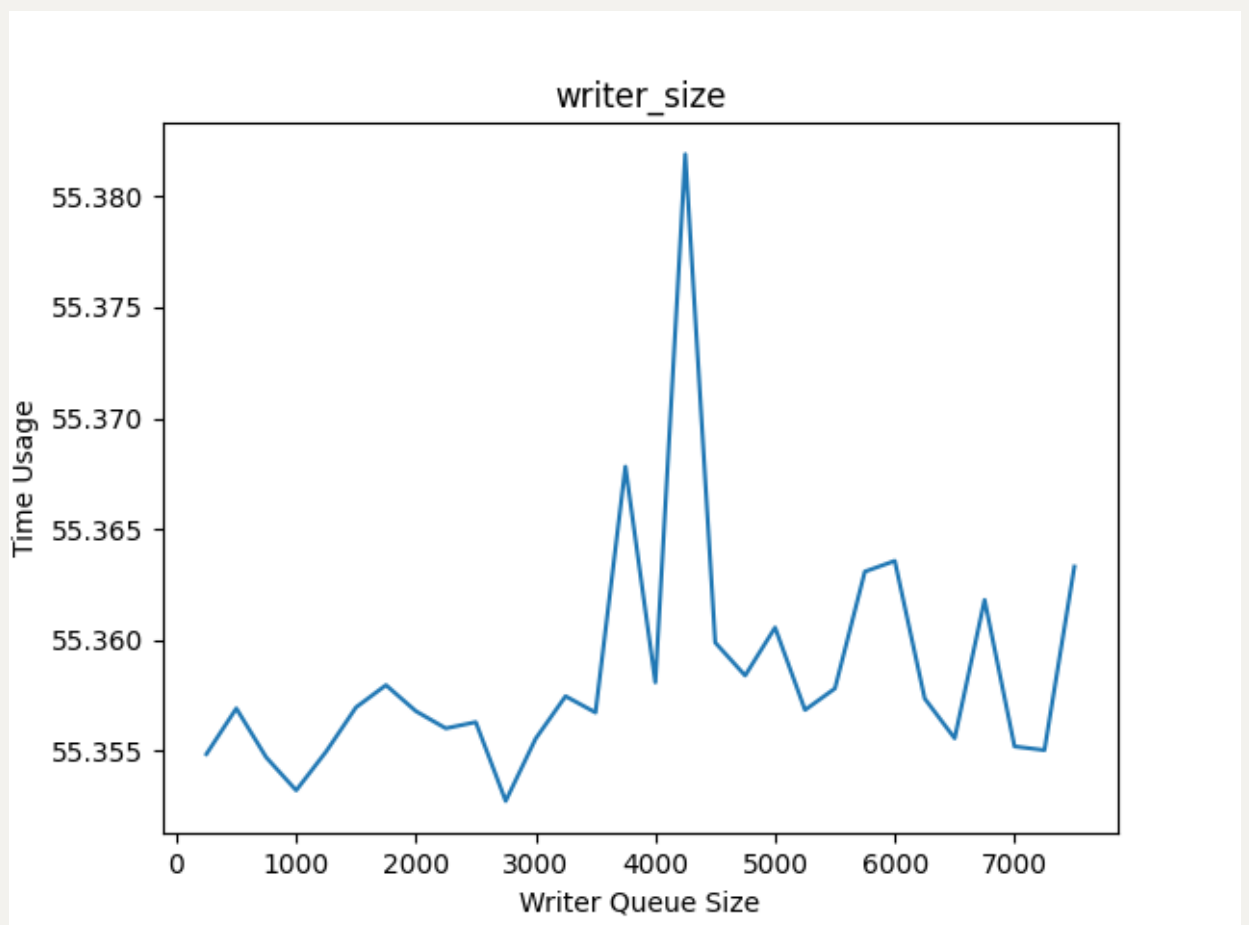
```

```

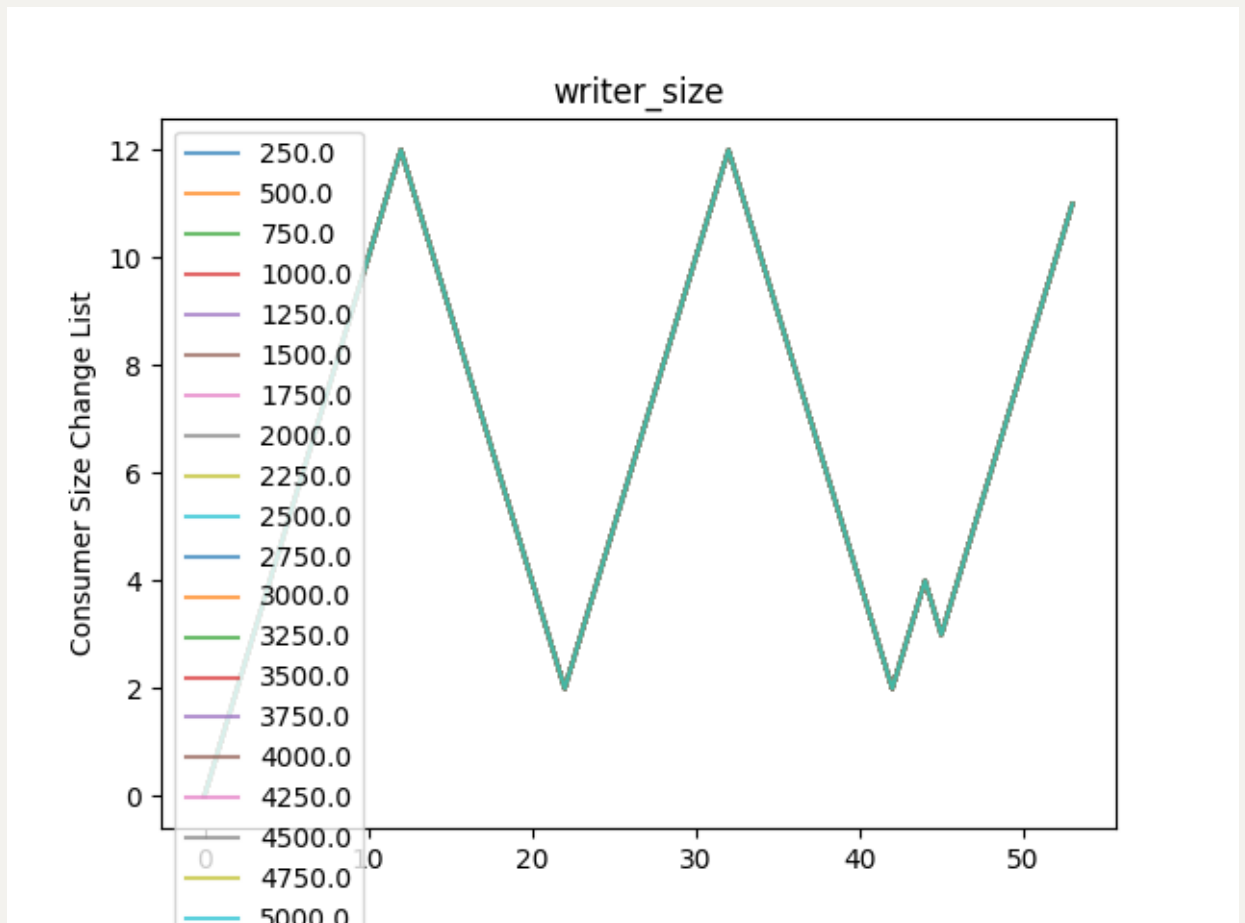
175         10., 11.])
176     ]
177     [7500.0 55.3633
178     array([ 0.,  1.,  2.,  3.,  4.,  5.,  6.,  7.,  8.,  9., 10.,
179            11., 12.,
180            11., 10.,  9.,  8.,  7.,  6.,  5.,  4.,  3.,  2.,  3.,
181            4.,  5.,
182            6.,  7.,  8.,  9., 10., 11., 12., 11., 10.,  9.,  8.,
183            7.,  6.,
184            5.,  4.,  3.,  2.,  3.,  4.,  3.,  4.,  5.,  6.,  7.,
185            8.,  9.,
186            10., 11.])
187     ]]

```

1. 圖像化 WRITER_QUEUE_SIZE 對 Time Usage 的影響:



2. 圖像化 WRITER_QUEUE_SIZE 對 CONSUMER SIZE CHANGE LIST 的影響:



3. 討論:

- a. WRITE_QUEUE_SIZE 和 Time Usage 的關係呈現中間有一高峰的情況, 但雖然圖上有一高峰, 但實際值並沒有差到太多 (約 0.03), 所以我認為 WRITER_QUEUE_SIZE 的大小對 Time Usage 影響不大
- b. 從圖可以發現 WRITER_QUEUE_SIZE 對 CONSUMER SIZE CHANGE 並沒有影響, 所有的線重疊在一起, 總體波形呈現谷峰谷峰谷峰
- c. What happens if WRITER_QUEUE_SIZE is very small?
 - i. 從這裡的結果來看, 對 Time Usage 與 CONSUMER SIZE CHANGE 並沒有太大影響

What happens if READER_QUEUE_SIZE is very small?

► READER_QUEUE_SIZE 由小到大所產生的 data:

```

1  ['Reader Queue Size', 'Time Usage', 'Consumer Size Change List']
2  [[5.0 55.3581
3     array([ 0.,  1.,  2.,  3.,  4.,  5.,  6.,  7.,  8.,  9., 10.,
            11., 12.,
```

```
4         11., 10., 9., 8., 7., 6., 5., 4., 3., 2., 3.,
      4., 5.,
5         6., 7., 8., 9., 10., 11., 12., 11., 10., 9., 8.,
      7., 6.,
6         5., 4., 3., 2., 3., 4., 3., 4., 5., 6., 7.,
      8., 9.,
7         10., 11.])
      ]
8 [10.0 55.3555375
9  array([ 0.,  1.,  2.,  3.,  4.,  5.,  6.,  7.,  8.,  9., 10.,
10 11., 12.,
11         11., 10., 9., 8., 7., 6., 5., 4., 3., 2., 3.,
      4., 5.,
12         6., 7., 8., 9., 10., 11., 12., 11., 10., 9., 8.,
      7., 6.,
13         5., 4., 3., 2., 3., 4., 3., 4., 5., 6., 7.,
      8., 9.,
14         10., 11.])
      ]
14 [15.0 55.363516666666667
15  array([ 0.,  1.,  2.,  3.,  4.,  5.,  6.,  7.,  8.,  9., 10.,
16 11., 12.,
17         11., 10., 9., 8., 7., 6., 5., 4., 3., 2., 3.,
      4., 5.,
18         6., 7., 8., 9., 10., 11., 12., 11., 10., 9., 8.,
      7., 6.,
19         5., 4., 3., 2., 3., 4., 3., 4., 5., 6., 7.,
      8., 9.,
20         10., 11.])
      ]
20 [20.0 55.35305714285714
21  array([ 0.,  1.,  2.,  3.,  4.,  5.,  6.,  7.,  8.,  9., 10.,
22 11., 12.,
23         11., 10., 9., 8., 7., 6., 5., 4., 3., 2., 3.,
      4., 5.,
24         6., 7., 8., 9., 10., 11., 12., 11., 10., 9., 8.,
      7., 6.,
25         5., 4., 3., 2., 3., 4., 3., 4., 5., 6., 7.,
      8., 9.,
```

```

25         10., 11.))
26     ]
27     [25.0 55.357883333333334
28     array([ 0.,  1.,  2.,  3.,  4.,  5.,  6.,  7.,  8.,  9., 10.,
11. 11., 12.,
29           11., 10.,  9.,  8.,  7.,  6.,  5.,  4.,  3.,  2.,  3.,
30           4.,  5.,
31           6.,  7.,  8.,  9., 10., 11., 12., 11., 10.,  9.,  8.,
32           7.,  6.,
33           5.,  4.,  3.,  2.,  3.,  4.,  3.,  4.,  5.,  6.,  7.,
34           8.,  9.,
35           10., 11.))
36     ]
37     [30.0 55.35285
38     array([ 0.,  1.,  2.,  3.,  4.,  5.,  6.,  7.,  8.,  9., 10.,
39 11., 12.,
40           11., 10.,  9.,  8.,  7.,  6.,  5.,  4.,  3.,  2.,  3.,
41           4.,  5.,
42           6.,  7.,  8.,  9., 10., 11., 12., 11., 10.,  9.,  8.,
43           7.,  6.,
44           5.,  4.,  3.,  2.,  3.,  4.,  3.,  4.,  5.,  6.,  7.,
45           8.,  9.,
46           10., 11.))
47     ]
48     [35.0 55.3582875
49     array([ 0.,  1.,  2.,  3.,  4.,  5.,  6.,  7.,  8.,  9., 10.,
50 11., 12.,
51           11., 10.,  9.,  8.,  7.,  6.,  5.,  4.,  3.,  2.,  3.,
52           4.,  5.,
53           6.,  7.,  8.,  9., 10., 11., 12., 11., 10.,  9.,  8.,
54           7.,  6.,
55           5.,  4.,  3.,  2.,  3.,  4.,  3.,  4.,  5.,  6.,  7.,
56           8.,  9.,
57           10., 11.))
58     ]
59     [40.0 55.35725714285714
60     array([ 0.,  1.,  2.,  3.,  4.,  5.,  6.,  7.,  8.,  9., 10.,
61 11., 12.,
62           11., 10.,  9.,  8.,  7.,  6.,  5.,  4.,  3.,  2.,  3.,
63           4.,  5.,
64           6.,  7.,  8.,  9., 10., 11., 12., 11., 10.,  9.,  8.,
65           7.,  6.,
66           5.,  4.,  3.,  2.,  3.,  4.,  3.,  4.,  5.,  6.,  7.,
67           8.,  9.,
68           10., 11.))
69     ]
70     [45.0 55.35625714285714
71     array([ 0.,  1.,  2.,  3.,  4.,  5.,  6.,  7.,  8.,  9., 10.,
72 11., 12.,
73           11., 10.,  9.,  8.,  7.,  6.,  5.,  4.,  3.,  2.,  3.,
74           4.,  5.,
75           6.,  7.,  8.,  9., 10., 11., 12., 11., 10.,  9.,  8.,
76           7.,  6.,
77           5.,  4.,  3.,  2.,  3.,  4.,  3.,  4.,  5.,  6.,  7.,
78           8.,  9.,
79           10., 11.))
80     ]
81     [50.0 55.35525714285714
82     array([ 0.,  1.,  2.,  3.,  4.,  5.,  6.,  7.,  8.,  9., 10.,
83 11., 12.,
84           11., 10.,  9.,  8.,  7.,  6.,  5.,  4.,  3.,  2.,  3.,
85           4.,  5.,
86           6.,  7.,  8.,  9., 10., 11., 12., 11., 10.,  9.,  8.,
87           7.,  6.,
88           5.,  4.,  3.,  2.,  3.,  4.,  3.,  4.,  5.,  6.,  7.,
89           8.,  9.,
90           10., 11.))
91     ]
92     [55.0 55.35425714285714
93     array([ 0.,  1.,  2.,  3.,  4.,  5.,  6.,  7.,  8.,  9., 10.,
94 11., 12.,
95           11., 10.,  9.,  8.,  7.,  6.,  5.,  4.,  3.,  2.,  3.,
96           4.,  5.,
97           6.,  7.,  8.,  9., 10., 11., 12., 11., 10.,  9.,  8.,
98           7.,  6.,
99           5.,  4.,  3.,  2.,  3.,  4.,  3.,  4.,  5.,  6.,  7.,
100          8.,  9.,
101          10., 11.))
102     ]
103     [60.0 55.35325714285714
104     array([ 0.,  1.,  2.,  3.,  4.,  5.,  6.,  7.,  8.,  9., 10.,
105 11., 12.,
106          11., 10.,  9.,  8.,  7.,  6.,  5.,  4.,  3.,  2.,  3.,
107          4.,  5.,
108          6.,  7.,  8.,  9., 10., 11., 12., 11., 10.,  9.,  8.,
109          7.,  6.,
110          5.,  4.,  3.,  2.,  3.,  4.,  3.,  4.,  5.,  6.,  7.,
111          8.,  9.,
112          10., 11.))
113     ]
114     [65.0 55.35225714285714
115     array([ 0.,  1.,  2.,  3.,  4.,  5.,  6.,  7.,  8.,  9., 10.,
116 11., 12.,
117          11., 10.,  9.,  8.,  7.,  6.,  5.,  4.,  3.,  2.,  3.,
118          4.,  5.,
119          6.,  7.,  8.,  9., 10., 11., 12., 11., 10.,  9.,  8.,
120          7.,  6.,
121          5.,  4.,  3.,  2.,  3.,  4.,  3.,  4.,  5.,  6.,  7.,
122          8.,  9.,
123          10., 11.))
124     ]
125     [70.0 55.35125714285714
126     array([ 0.,  1.,  2.,  3.,  4.,  5.,  6.,  7.,  8.,  9., 10.,
127 11., 12.,
128          11., 10.,  9.,  8.,  7.,  6.,  5.,  4.,  3.,  2.,  3.,
129          4.,  5.,
130          6.,  7.,  8.,  9., 10., 11., 12., 11., 10.,  9.,  8.,
131          7.,  6.,
132          5.,  4.,  3.,  2.,  3.,  4.,  3.,  4.,  5.,  6.,  7.,
133          8.,  9.,
134          10., 11.))
135     ]
136     [75.0 55.35025714285714
137     array([ 0.,  1.,  2.,  3.,  4.,  5.,  6.,  7.,  8.,  9., 10.,
138 11., 12.,
139          11., 10.,  9.,  8.,  7.,  6.,  5.,  4.,  3.,  2.,  3.,
140          4.,  5.,
141          6.,  7.,  8.,  9., 10., 11., 12., 11., 10.,  9.,  8.,
142          7.,  6.,
143          5.,  4.,  3.,  2.,  3.,  4.,  3.,  4.,  5.,  6.,  7.,
144          8.,  9.,
145          10., 11.))
146     ]
147     [80.0 55.34925714285714
148     array([ 0.,  1.,  2.,  3.,  4.,  5.,  6.,  7.,  8.,  9., 10.,
149 11., 12.,
150          11., 10.,  9.,  8.,  7.,  6.,  5.,  4.,  3.,  2.,  3.,
151          4.,  5.,
152          6.,  7.,  8.,  9., 10., 11., 12., 11., 10.,  9.,  8.,
153          7.,  6.,
154          5.,  4.,  3.,  2.,  3.,  4.,  3.,  4.,  5.,  6.,  7.,
155          8.,  9.,
156          10., 11.))
157     ]
158     [85.0 55.34825714285714
159     array([ 0.,  1.,  2.,  3.,  4.,  5.,  6.,  7.,  8.,  9., 10.,
160 11., 12.,
161          11., 10.,  9.,  8.,  7.,  6.,  5.,  4.,  3.,  2.,  3.,
162          4.,  5.,
163          6.,  7.,  8.,  9., 10., 11., 12., 11., 10.,  9.,  8.,
164          7.,  6.,
165          5.,  4.,  3.,  2.,  3.,  4.,  3.,  4.,  5.,  6.,  7.,
166          8.,  9.,
167          10., 11.))
168     ]
169     [90.0 55.34725714285714
170     array([ 0.,  1.,  2.,  3.,  4.,  5.,  6.,  7.,  8.,  9., 10.,
171 11., 12.,
172          11., 10.,  9.,  8.,  7.,  6.,  5.,  4.,  3.,  2.,  3.,
173          4.,  5.,
174          6.,  7.,  8.,  9., 10., 11., 12., 11., 10.,  9.,  8.,
175          7.,  6.,
176          5.,  4.,  3.,  2.,  3.,  4.,  3.,  4.,  5.,  6.,  7.,
177          8.,  9.,
178          10., 11.))
179     ]
180     [95.0 55.34625714285714
181     array([ 0.,  1.,  2.,  3.,  4.,  5.,  6.,  7.,  8.,  9., 10.,
182 11., 12.,
```

```
46         11., 10., 9., 8., 7., 6., 5., 4., 3., 2., 3.,
47         4., 5.,
48         6., 7., 8., 9., 10., 11., 12., 11., 10., 9., 8.,
49         7., 6.,
50         5., 4., 3., 2., 3., 4., 3., 4., 5., 6., 7.,
51         8., 9.,
52         10., 11.])
53     ]
54     [45.0 55.35682857142857
55     array([ 0., 1., 2., 3., 4., 5., 6., 7., 8., 9., 10.,
56            11., 12.,
57            11., 10., 9., 8., 7., 6., 5., 4., 3., 2., 3.,
58            4., 5.,
59            6., 7., 8., 9., 10., 11., 12., 11., 10., 9., 8.,
60            7., 6.,
61            5., 4., 3., 2., 3., 4., 3., 4., 5., 6., 7.,
62            8., 9.,
63            10., 11.])
64     ]
65     [50.0 55.356166666666667
66     array([ 0., 1., 2., 3., 4., 5., 6., 7., 8., 9., 10.,
67            11., 12.,
68            11., 10., 9., 8., 7., 6., 5., 4., 3., 2., 3.,
69            4., 5.,
70            6., 7., 8., 9., 10., 11., 12., 11., 10., 9., 8.,
71            7., 6.,
72            5., 4., 3., 2., 3., 4., 3., 4., 5., 6., 7.,
73            8., 9.,
74            10., 11.])
75     ]
76     [55.0 55.360825000000006
77     array([ 0., 1., 2., 3., 4., 5., 6., 7., 8., 9., 10.,
78            11., 12.,
79            11., 10., 9., 8., 7., 6., 5., 4., 3., 2., 3.,
80            4., 5.,
81            6., 7., 8., 9., 10., 11., 12., 11., 10., 9., 8.,
82            7., 6.,
83            5., 4., 3., 2., 3., 4., 3., 4., 5., 6., 7.,
84            8., 9.,
85            10., 11.])
86     ]
```



```

67         10., 11.])
68     ]
69     [60.0 55.3538
70     array([ 0.,  1.,  2.,  3.,  4.,  5.,  6.,  7.,  8.,  9., 10.,
71            11., 12.,
72            11., 10.,  9.,  8.,  7.,  6.,  5.,  4.,  3.,  2.,  3.,
73            4.,  5.,
74            6.,  7.,  8.,  9., 10., 11., 12., 11., 10.,  9.,  8.,
75            7.,  6.,
76            5.,  4.,  3.,  2.,  3.,  4.,  3.,  4.,  5.,  6.,  7.,
77            8.,  9.,
78            10., 11.])
79     ]
80     [65.0 55.358371428571424
81     array([ 0.,  1.,  2.,  3.,  4.,  5.,  6.,  7.,  8.,  9., 10.,
82            11., 12.,
83            11., 10.,  9.,  8.,  7.,  6.,  5.,  4.,  3.,  2.,  3.,
84            4.,  5.,
85            6.,  7.,  8.,  9., 10., 11., 12., 11., 10.,  9.,  8.,
86            7.,  6.,
87            5.,  4.,  3.,  2.,  3.,  4.,  3.,  4.,  5.,  6.,  7.,
88            8.,  9.,
89            10., 11.])
90     ]
91     [70.0 55.354685714285715
92     array([ 0.,  1.,  2.,  3.,  4.,  5.,  6.,  7.,  8.,  9., 10.,
93            11., 12.,
94            11., 10.,  9.,  8.,  7.,  6.,  5.,  4.,  3.,  2.,  3.,
95            4.,  5.,
96            6.,  7.,  8.,  9., 10., 11., 12., 11., 10.,  9.,  8.,
97            7.,  6.,
98            5.,  4.,  3.,  2.,  3.,  4.,  3.,  4.,  5.,  6.,  7.,
99            8.,  9.,
100           10., 11.])
101     ]
102     [75.0 55.35552857142857
103     array([ 0.,  1.,  2.,  3.,  4.,  5.,  6.,  7.,  8.,  9., 10.,
104            11., 12.,

```

```
88         11., 10., 9., 8., 7., 6., 5., 4., 3., 2., 3.,
      4., 5.,
89         6., 7., 8., 9., 10., 11., 12., 11., 10., 9., 8.,
      7., 6.,
90         5., 4., 3., 2., 3., 4., 3., 4., 5., 6., 7.,
      8., 9.,
91         10., 11.])
      ]
92 [80.0 55.357275
93  array([ 0.,  1.,  2.,  3.,  4.,  5.,  6.,  7.,  8.,  9., 10.,
      11., 12.,
94         11., 10., 9., 8., 7., 6., 5., 4., 3., 2., 3.,
      4., 5.,
95         6., 7., 8., 9., 10., 11., 12., 11., 10., 9., 8.,
      7., 6.,
96         5., 4., 3., 2., 3., 4., 3., 4., 5., 6., 7.,
      8., 9.,
97         10., 11.])
      ]
98 [85.0 55.35604285714287
99  array([ 0.,  1.,  2.,  3.,  4.,  5.,  6.,  7.,  8.,  9., 10.,
      11., 12.,
100         11., 10., 9., 8., 7., 6., 5., 4., 3., 2., 3.,
      4., 5.,
101         6., 7., 8., 9., 10., 11., 12., 11., 10., 9., 8.,
      7., 6.,
102         5., 4., 3., 2., 3., 4., 3., 4., 5., 6., 7.,
      8., 9.,
103         10., 11.])
      ]
104 [90.0 55.35945714285714
105  array([ 0.,  1.,  2.,  3.,  4.,  5.,  6.,  7.,  8.,  9., 10.,
      11., 12.,
106         11., 10., 9., 8., 7., 6., 5., 4., 3., 2., 3.,
      4., 5.,
107         6., 7., 8., 9., 10., 11., 12., 11., 10., 9., 8.,
      7., 6.,
108         5., 4., 3., 2., 3., 4., 3., 4., 5., 6., 7.,
      8., 9.,
```

```

109         10., 11.])
110     ]
111     [95.0 55.3556750000000005
112     array([ 0.,  1.,  2.,  3.,  4.,  5.,  6.,  7.,  8.,  9., 10.,
113 11., 12.,
114           11., 10.,  9.,  8.,  7.,  6.,  5.,  4.,  3.,  2.,  3.,
115 4.,  5.,
116           6.,  7.,  8.,  9., 10., 11., 12., 11., 10.,  9.,  8.,
117 7.,  6.,
118           5.,  4.,  3.,  2.,  3.,  4.,  3.,  4.,  5.,  6.,  7.,
119 8.,  9.,
120           10., 11.])
121     ]
122     [100.0 55.35617142857143
123     array([ 0.,  1.,  2.,  3.,  4.,  5.,  6.,  7.,  8.,  9., 10.,
124 11., 12.,
125           11., 10.,  9.,  8.,  7.,  6.,  5.,  4.,  3.,  2.,  3.,
126 4.,  5.,
127           6.,  7.,  8.,  9., 10., 11., 12., 11., 10.,  9.,  8.,
128 7.,  6.,
129           5.,  4.,  3.,  2.,  3.,  4.,  3.,  4.,  5.,  6.,  7.,
130 8.,  9.,
131           10., 11.])
132     ]
133     [105.0 55.35452857142857
134     array([ 0.,  1.,  2.,  3.,  4.,  5.,  6.,  7.,  8.,  9., 10.,
135 11., 12.,
136           11., 10.,  9.,  8.,  7.,  6.,  5.,  4.,  3.,  2.,  3.,
137 4.,  5.,
138           6.,  7.,  8.,  9., 10., 11., 12., 11., 10.,  9.,  8.,
139 7.,  6.,
140           5.,  4.,  3.,  2.,  3.,  4.,  3.,  4.,  5.,  6.,  7.,
141 8.,  9.,
142           10., 11.])
143     ]
144     [110.0 55.354650000000001
145     array([ 0.,  1.,  2.,  3.,  4.,  5.,  6.,  7.,  8.,  9., 10.,
146 11., 12.,

```

```
130         11., 10., 9., 8., 7., 6., 5., 4., 3., 2., 3.,
131         4., 5.,
132         6., 7., 8., 9., 10., 11., 12., 11., 10., 9., 8.,
133         7., 6.,
134         5., 4., 3., 2., 3., 4., 3., 4., 5., 6., 7.,
135         8., 9.,
136         10., 11.])
137     ]
138     [115.0 55.356971428571434
139     array([ 0., 1., 2., 3., 4., 5., 6., 7., 8., 9., 10.,
140            11., 12.,
141            11., 10., 9., 8., 7., 6., 5., 4., 3., 2., 3.,
142            4., 5.,
143            6., 7., 8., 9., 10., 11., 12., 11., 10., 9., 8.,
144            7., 6.,
145            5., 4., 3., 2., 3., 4., 3., 4., 5., 6., 7.,
146            8., 9.,
147            10., 11.])
148     ]
149     [120.0 55.35452857142857
150     array([ 0., 1., 2., 3., 4., 5., 6., 7., 8., 9., 10.,
151            11., 12.,
152            11., 10., 9., 8., 7., 6., 5., 4., 3., 2., 3.,
153            4., 5.,
154            6., 7., 8., 9., 10., 11., 12., 11., 10., 9., 8.,
155            7., 6.,
156            5., 4., 3., 2., 3., 4., 3., 4., 5., 6., 7.,
157            8., 9.,
158            10., 11.])]
```

```

151 [130.0 53.8235
152   array([ 0.,  1.,  2.,  3.,  4.,  5.,  6.,  7.,  8.,  9., 10.,
153         11., 12.,
154           11., 10.,  9.,  8.,  7.,  6.,  5.,  4.,  3.,  2.,  3.,
155           4.,  5.,
156           6.,  7.,  8.,  9., 10., 11., 12., 11., 10.,  9.,  8.,
157           7.,  6.,
158           5.,  4.,  3.,  2.,  3.,  4.,  5.,  6.,  7.,  8.,  9.,
159           10., 11.])]
160 [135.0 55.362728571428576
161   array([ 0.,  1.,  2.,  3.,  4.,  5.,  6.,  7.,  8.,  9., 10.,
162         11., 12.,
163           11., 10.,  9.,  8.,  7.,  6.,  5.,  4.,  3.,  2.,  3.,
164           4.,  5.,
165           6.,  7.,  8.,  9., 10., 11., 12., 11., 10.,  9.,  8.,
166           7.,  6.,
167           5.,  4.,  3.,  2.,  3.,  4.,  3.,  4.,  5.,  6.,  7.,
168           8.,  9.,
169           10., 11.])]
170 [140.0 55.35525714285715
171   array([ 0.,  1.,  2.,  3.,  4.,  5.,  6.,  7.,  8.,  9., 10.,
172         11., 12.,
173           11., 10.,  9.,  8.,  7.,  6.,  5.,  4.,  3.,  2.,  3.,
174           4.,  5.,
175           6.,  7.,  8.,  9., 10., 11., 12., 11., 10.,  9.,  8.,
176           7.,  6.,
177           5.,  4.,  3.,  2.,  3.,  4.,  3.,  4.,  5.,  6.,  7.,
178           8.,  9.,
179           10., 11.])]
180 [145.0 55.363487500000005
181   array([ 0.,  1.,  2.,  3.,  4.,  5.,  6.,  7.,  8.,  9., 10.,
182         11., 12.,
183           11., 10.,  9.,  8.,  7.,  6.,  5.,  4.,  3.,  2.,  3.,
184           4.,  5.,
185           6.,  7.,  8.,  9., 10., 11., 12., 11., 10.,  9.,  8.,
186           7.,  6.,
187           5.,  4.,  3.,  2.,  3.,  4.,  3.,  4.,  5.,  6.,  7.,
188           8.,  9.,
189           10., 11.])]

```

```
172         5., 4., 3., 2., 3., 4., 3., 4., 5., 6., 7.,
173         8., 9.,
174         10., 11.])
175     ]
176     [150.0 53.826074999999996
177     array([ 0., 1., 2., 3., 4., 5., 6., 7., 8., 9., 10.,
178            11., 12.,
179            11., 10., 9., 8., 7., 6., 5., 4., 3., 2., 3.,
180            4., 5.,
181            6., 7., 8., 9., 10., 11., 12., 11., 10., 9., 8.,
182            7., 6.,
183            5., 4., 3., 2., 3., 4., 5., 6., 7., 8., 9.,
184            10., 11.])]
185     [155.0 55.354471428571436
186     array([ 0., 1., 2., 3., 4., 5., 6., 7., 8., 9., 10.,
187            11., 12.,
188            11., 10., 9., 8., 7., 6., 5., 4., 3., 2., 3.,
189            4., 5.,
190            6., 7., 8., 9., 10., 11., 12., 11., 10., 9., 8.,
191            7., 6.,
192            5., 4., 3., 2., 3., 4., 3., 4., 5., 6., 7.,
193            8., 9.,
194            10., 11.])]
195     ]
196     [160.0 55.35376
197     array([ 0., 1., 2., 3., 4., 5., 6., 7., 8., 9., 10.,
198            11., 12.,
199            11., 10., 9., 8., 7., 6., 5., 4., 3., 2., 3.,
200            4., 5.,
201            6., 7., 8., 9., 10., 11., 12., 11., 10., 9., 8.,
202            7., 6.,
203            5., 4., 3., 2., 3., 4., 3., 4., 5., 6., 7.,
204            8., 9.,
205            10., 11.])]
206     ]
207     [165.0 55.358233333333324
208     array([ 0., 1., 2., 3., 4., 5., 6., 7., 8., 9., 10.,
209            11., 12.,
```

```
193         11., 10., 9., 8., 7., 6., 5., 4., 3., 2., 3.,
194         4., 5.,
195         6., 7., 8., 9., 10., 11., 12., 11., 10., 9., 8.,
196         7., 6.,
197         5., 4., 3., 2., 3., 4., 3., 4., 5., 6., 7.,
198         8., 9.,
199         10., 11.])
200     ]
201     [170.0 55.35251666666667
202     array([ 0., 1., 2., 3., 4., 5., 6., 7., 8., 9., 10.,
203            11., 12.,
204            11., 10., 9., 8., 7., 6., 5., 4., 3., 2., 3.,
205            4., 5.,
206            6., 7., 8., 9., 10., 11., 12., 11., 10., 9., 8.,
207            7., 6.,
208            5., 4., 3., 2., 3., 4., 3., 4., 5., 6., 7.,
209            8., 9.,
210            10., 11.])
211     ]
212     [175.0 55.355740000000004
213     array([ 0., 1., 2., 3., 4., 5., 6., 7., 8., 9., 10.,
214            11., 12.,
215            11., 10., 9., 8., 7., 6., 5., 4., 3., 2., 3.,
216            4., 5.,
217            6., 7., 8., 9., 10., 11., 12., 11., 10., 9., 8.,
218            7., 6.,
219            5., 4., 3., 2., 3., 4., 3., 4., 5., 6., 7.,
220            8., 9.,
221            10., 11.])
222     ]
223     [180.0 55.35517142857144
224     array([ 0., 1., 2., 3., 4., 5., 6., 7., 8., 9., 10.,
225            11., 12.,
226            11., 10., 9., 8., 7., 6., 5., 4., 3., 2., 3.,
227            4., 5.,
228            6., 7., 8., 9., 10., 11., 12., 11., 10., 9., 8.,
229            7., 6.,
230            5., 4., 3., 2., 3., 4., 3., 4., 5., 6., 7.,
231            8., 9.,
232            10., 11.])
233     ]
```

```

214         10., 11.])
215     ]
216     [185.0 55.35237142857142
217     array([ 0.,  1.,  2.,  3.,  4.,  5.,  6.,  7.,  8.,  9., 10.,
218            11., 12.,
219            11., 10.,  9.,  8.,  7.,  6.,  5.,  4.,  3.,  2.,  3.,
220            4.,  5.,
221            6.,  7.,  8.,  9., 10., 11., 12., 11., 10.,  9.,  8.,
222            7.,  6.,
223            5.,  4.,  3.,  2.,  3.,  4.,  3.,  4.,  5.,  6.,  7.,
224            8.,  9.,
225            10., 11.])
226     ]
227     [190.0 55.3581
228     array([ 0.,  1.,  2.,  3.,  4.,  5.,  6.,  7.,  8.,  9., 10.,
229            11., 12.,
230            11., 10.,  9.,  8.,  7.,  6.,  5.,  4.,  3.,  2.,  3.,
231            4.,  5.,
232            6.,  7.,  8.,  9., 10., 11., 12., 11., 10.,  9.,  8.,
233            7.,  6.,
234            5.,  4.,  3.,  2.,  3.,  4.,  3.,  4.,  5.,  6.,  7.,
235            8.,  9.,
236            10., 11.])
237     ]
238     [195.0 55.357442857142864
239     array([ 0.,  1.,  2.,  3.,  4.,  5.,  6.,  7.,  8.,  9., 10.,
240            11., 12.,
241            11., 10.,  9.,  8.,  7.,  6.,  5.,  4.,  3.,  2.,  3.,
242            4.,  5.,
243            6.,  7.,  8.,  9., 10., 11., 12., 11., 10.,  9.,  8.,
244            7.,  6.,
245            5.,  4.,  3.,  2.,  3.,  4.,  3.,  4.,  5.,  6.,  7.,
246            8.,  9.,
247            10., 11.])
248     ]
249     [200.0 55.35788333333333
250     array([ 0.,  1.,  2.,  3.,  4.,  5.,  6.,  7.,  8.,  9., 10.,
251            11., 12.,
252            11., 10.,  9.,  8.,  7.,  6.,  5.,  4.,  3.,  2.,  3.,
253            4.,  5.,
254            6.,  7.,  8.,  9., 10., 11., 12., 11., 10.,  9.,  8.,
255            7.,  6.,
256            5.,  4.,  3.,  2.,  3.,  4.,  3.,  4.,  5.,  6.,  7.,
257            8.,  9.,
258            10., 11.])
259     ]

```



```
235         11., 10., 9., 8., 7., 6., 5., 4., 3., 2., 3.,
      4., 5.,
236         6., 7., 8., 9., 10., 11., 12., 11., 10., 9., 8.,
      7., 6.,
237         5., 4., 3., 2., 3., 4., 3., 4., 5., 6., 7.,
      8., 9.,
238         10., 11.])
      ]
239 [205.0 55.35902857142857
240  array([ 0.,  1.,  2.,  3.,  4.,  5.,  6.,  7.,  8.,  9., 10.,
      11., 12.,
241         11., 10., 9., 8., 7., 6., 5., 4., 3., 2., 3.,
      4., 5.,
242         6., 7., 8., 9., 10., 11., 12., 11., 10., 9., 8.,
      7., 6.,
243         5., 4., 3., 2., 3., 4., 3., 4., 5., 6., 7.,
      8., 9.,
244         10., 11.])
      ]
245 [210.0 55.355114285714286
246  array([ 0.,  1.,  2.,  3.,  4.,  5.,  6.,  7.,  8.,  9., 10.,
      11., 12.,
247         11., 10., 9., 8., 7., 6., 5., 4., 3., 2., 3.,
      4., 5.,
248         6., 7., 8., 9., 10., 11., 12., 11., 10., 9., 8.,
      7., 6.,
249         5., 4., 3., 2., 3., 4., 3., 4., 5., 6., 7.,
      8., 9.,
250         10., 11.])
      ]
251 [215.0 55.3632
252  array([ 0.,  1.,  2.,  3.,  4.,  5.,  6.,  7.,  8.,  9., 10.,
      11., 12.,
253         11., 10., 9., 8., 7., 6., 5., 4., 3., 2., 3.,
      4., 5.,
254         6., 7., 8., 9., 10., 11., 12., 11., 10., 9., 8.,
      7., 6.,
255         5., 4., 3., 2., 3., 4., 3., 4., 5., 6., 7.,
      8., 9.,
```

```

256         10., 11.])
257     ]
258     [220.0 55.35581428571429
259     array([ 0.,  1.,  2.,  3.,  4.,  5.,  6.,  7.,  8.,  9., 10.,
260            11., 12.,
261            11., 10.,  9.,  8.,  7.,  6.,  5.,  4.,  3.,  2.,  3.,
262            4.,  5.,
263            6.,  7.,  8.,  9., 10., 11., 12., 11., 10.,  9.,  8.,
264            7.,  6.,
265            5.,  4.,  3.,  2.,  3.,  4.,  3.,  4.,  5.,  6.,  7.,
266            8.,  9.,
267            10., 11.])
268     ]
269     [225.0 55.35664285714285
270     array([ 0.,  1.,  2.,  3.,  4.,  5.,  6.,  7.,  8.,  9., 10.,
271            11., 12.,
272            11., 10.,  9.,  8.,  7.,  6.,  5.,  4.,  3.,  2.,  3.,
273            4.,  5.,
274            6.,  7.,  8.,  9., 10., 11., 12., 11., 10.,  9.,  8.,
275            7.,  6.,
276            5.,  4.,  3.,  2.,  3.,  4.,  3.,  4.,  5.,  6.,  7.,
277            8.,  9.,
278            10., 11.])
279     ]
280     [230.0 55.36134285714286
281     array([ 0.,  1.,  2.,  3.,  4.,  5.,  6.,  7.,  8.,  9., 10.,
282            11., 12.,
283            11., 10.,  9.,  8.,  7.,  6.,  5.,  4.,  3.,  2.,  3.,
284            4.,  5.,
285            6.,  7.,  8.,  9., 10., 11., 12., 11., 10.,  9.,  8.,
286            7.,  6.,
287            5.,  4.,  3.,  2.,  3.,  4.,  3.,  4.,  5.,  6.,  7.,
288            8.,  9.,
289            10., 11.])
290     ]
291     [235.0 55.36058333333333
292     array([ 0.,  1.,  2.,  3.,  4.,  5.,  6.,  7.,  8.,  9., 10.,
293            11., 12.,

```

```
277         11., 10., 9., 8., 7., 6., 5., 4., 3., 2., 3.,
        4., 5.,
278         6., 7., 8., 9., 10., 11., 12., 11., 10., 9., 8.,
        7., 6.,
279         5., 4., 3., 2., 3., 4., 3., 4., 5., 6., 7.,
        8., 9.,
280         10., 11.])
        ]
281 [240.0 55.36203750000001
282  array([ 0.,  1.,  2.,  3.,  4.,  5.,  6.,  7.,  8.,  9., 10.,
        11., 12.,
283         11., 10., 9., 8., 7., 6., 5., 4., 3., 2., 3.,
        4., 5.,
284         6., 7., 8., 9., 10., 11., 12., 11., 10., 9., 8.,
        7., 6.,
285         5., 4., 3., 2., 3., 4., 3., 4., 5., 6., 7.,
        8., 9.,
286         10., 11.])
        ]
287 [245.0 55.356042857142846
288  array([ 0.,  1.,  2.,  3.,  4.,  5.,  6.,  7.,  8.,  9., 10.,
        11., 12.,
289         11., 10., 9., 8., 7., 6., 5., 4., 3., 2., 3.,
        4., 5.,
290         6., 7., 8., 9., 10., 11., 12., 11., 10., 9., 8.,
        7., 6.,
291         5., 4., 3., 2., 3., 4., 3., 4., 5., 6., 7.,
        8., 9.,
292         10., 11.])
        ]
293 [250.0 55.36475714285714
294  array([ 0.,  1.,  2.,  3.,  4.,  5.,  6.,  7.,  8.,  9., 10.,
        11., 12.,
295         11., 10., 9., 8., 7., 6., 5., 4., 3., 2., 3.,
        4., 5.,
296         6., 7., 8., 9., 10., 11., 12., 11., 10., 9., 8.,
        7., 6.,
297         5., 4., 3., 2., 3., 4., 3., 4., 5., 6., 7.,
        8., 9.,
```

```

298         10., 11.])
299     ]
300     [255.0 55.3564625
301     array([ 0.,  1.,  2.,  3.,  4.,  5.,  6.,  7.,  8.,  9., 10.,
302            11., 12.,
303            11., 10.,  9.,  8.,  7.,  6.,  5.,  4.,  3.,  2.,  3.,
304            4.,  5.,
305            6.,  7.,  8.,  9., 10., 11., 12., 11., 10.,  9.,  8.,
306            7.,  6.,
307            5.,  4.,  3.,  2.,  3.,  4.,  3.,  4.,  5.,  6.,  7.,
308            8.,  9.,
309            10., 11.])
310     ]
311     [260.0 55.35791428571429
312     array([ 0.,  1.,  2.,  3.,  4.,  5.,  6.,  7.,  8.,  9., 10.,
313            11., 12.,
314            11., 10.,  9.,  8.,  7.,  6.,  5.,  4.,  3.,  2.,  3.,
315            4.,  5.,
316            6.,  7.,  8.,  9., 10., 11., 12., 11., 10.,  9.,  8.,
317            7.,  6.,
318            5.,  4.,  3.,  2.,  3.,  4.,  3.,  4.,  5.,  6.,  7.,
319            8.,  9.,
320            10., 11.])
321     ]
322     [265.0 55.35582857142857
323     array([ 0.,  1.,  2.,  3.,  4.,  5.,  6.,  7.,  8.,  9., 10.,
324            11., 12.,
325            11., 10.,  9.,  8.,  7.,  6.,  5.,  4.,  3.,  2.,  3.,
326            4.,  5.,
327            6.,  7.,  8.,  9., 10., 11., 12., 11., 10.,  9.,  8.,
328            7.,  6.,
329            5.,  4.,  3.,  2.,  3.,  4.,  3.,  4.,  5.,  6.,  7.,
330            8.,  9.,
331            10., 11.])
332     ]
333     [270.0 55.362116666666666
334     array([ 0.,  1.,  2.,  3.,  4.,  5.,  6.,  7.,  8.,  9., 10.,
335            11., 12.,

```

```

319         11., 10., 9., 8., 7., 6., 5., 4., 3., 2., 3.,
        4., 5.,
320         6., 7., 8., 9., 10., 11., 12., 11., 10., 9., 8.,
        7., 6.,
321         5., 4., 3., 2., 3., 4., 3., 4., 5., 6., 7.,
        8., 9.,
322         10., 11.])
        ]
323 [275.0 55.360299999999995
324 array([ 0., 1., 2., 3., 4., 5., 6., 7., 8., 9., 10.,
        11., 12.,
325         11., 10., 9., 8., 7., 6., 5., 4., 3., 2., 3.,
        4., 5.,
326         6., 7., 8., 9., 10., 11., 12., 11., 10., 9., 8.,
        7., 6.,
327         5., 4., 3., 2., 3., 4., 3., 4., 5., 6., 7.,
        8., 9.,
328         10., 11.])
        ]
329 [280.0 55.35605714285714
330 array([ 0., 1., 2., 3., 4., 5., 6., 7., 8., 9., 10.,
        11., 12.,
331         11., 10., 9., 8., 7., 6., 5., 4., 3., 2., 3.,
        4., 5.,
332         6., 7., 8., 9., 10., 11., 12., 11., 10., 9., 8.,
        7., 6.,
333         5., 4., 3., 2., 3., 4., 3., 4., 5., 6., 7.,
        8., 9.,
334         10., 11.])
        ]
335 [285.0 55.35338
336 array([ 0., 1., 2., 3., 4., 5., 6., 7., 8., 9., 10.,
        11., 12.,
337         11., 10., 9., 8., 7., 6., 5., 4., 3., 2., 3.,
        4., 5.,
338         6., 7., 8., 9., 10., 11., 12., 11., 10., 9., 8.,
        7., 6.,
339         5., 4., 3., 2., 3., 4., 3., 4., 5., 6., 7.,
        8., 9.,

```

```

340         10., 11.])
341     ]
342     [290.0 55.35475714285714
343     array([ 0.,  1.,  2.,  3.,  4.,  5.,  6.,  7.,  8.,  9., 10.,
344            11., 12.,
345            11., 10.,  9.,  8.,  7.,  6.,  5.,  4.,  3.,  2.,  3.,
346            4.,  5.,
347            6.,  7.,  8.,  9., 10., 11., 12., 11., 10.,  9.,  8.,
348            7.,  6.,
349            5.,  4.,  3.,  2.,  3.,  4.,  3.,  4.,  5.,  6.,  7.,
350            8.,  9.,
351            10., 11.])]
352     ]
353     [295.0 55.35774
354     array([ 0.,  1.,  2.,  3.,  4.,  5.,  6.,  7.,  8.,  9., 10.,
355            11., 12.,
356            11., 10.,  9.,  8.,  7.,  6.,  5.,  4.,  3.,  2.,  3.,
357            4.,  5.,
358            6.,  7.,  8.,  9., 10., 11., 12., 11., 10.,  9.,  8.,
359            7.,  6.,
360            5.,  4.,  3.,  2.,  3.,  4.,  3.,  4.,  5.,  6.,  7.,
361            8.,  9.,
362            10., 11.])]
363     ]
364     [300.0 55.3563
365     array([ 0.,  1.,  2.,  3.,  4.,  5.,  6.,  7.,  8.,  9., 10.,
366            11., 12.,
367            11., 10.,  9.,  8.,  7.,  6.,  5.,  4.,  3.,  2.,  3.,
368            4.,  5.,
369            6.,  7.,  8.,  9., 10., 11., 12., 11., 10.,  9.,  8.,
370            7.,  6.,
371            5.,  4.,  3.,  2.,  3.,  4.,  3.,  4.,  5.,  6.,  7.,
372            8.,  9.,
373            10., 11.])]
374     ]
375     [305.0 55.36382857142856
376     array([ 0.,  1.,  2.,  3.,  4.,  5.,  6.,  7.,  8.,  9., 10.,
377            11., 12.,
378            11., 10.,  9.,  8.,  7.,  6.,  5.,  4.,  3.,  2.,  3.,
379            4.,  5.,
380            6.,  7.,  8.,  9., 10., 11., 12., 11., 10.,  9.,  8.,
381            7.,  6.,
382            5.,  4.,  3.,  2.,  3.,  4.,  3.,  4.,  5.,  6.,  7.,
383            8.,  9.,
384            10., 11.])]
385     ]
386     [310.0 55.36785714285714
387     array([ 0.,  1.,  2.,  3.,  4.,  5.,  6.,  7.,  8.,  9., 10.,
388            11., 12.,
389            11., 10.,  9.,  8.,  7.,  6.,  5.,  4.,  3.,  2.,  3.,
390            4.,  5.,
391            6.,  7.,  8.,  9., 10., 11., 12., 11., 10.,  9.,  8.,
392            7.,  6.,
393            5.,  4.,  3.,  2.,  3.,  4.,  3.,  4.,  5.,  6.,  7.,
394            8.,  9.,
395            10., 11.])]
396     ]
397     [315.0 55.37188571428571
398     array([ 0.,  1.,  2.,  3.,  4.,  5.,  6.,  7.,  8.,  9., 10.,
399            11., 12.,
400            11., 10.,  9.,  8.,  7.,  6.,  5.,  4.,  3.,  2.,  3.,
401            4.,  5.,
402            6.,  7.,  8.,  9., 10., 11., 12., 11., 10.,  9.,  8.,
403            7.,  6.,
404            5.,  4.,  3.,  2.,  3.,  4.,  3.,  4.,  5.,  6.,  7.,
405            8.,  9.,
406            10., 11.])]
407     ]
408     [320.0 55.37591428571428
409     array([ 0.,  1.,  2.,  3.,  4.,  5.,  6.,  7.,  8.,  9., 10.,
410            11., 12.,
411            11., 10.,  9.,  8.,  7.,  6.,  5.,  4.,  3.,  2.,  3.,
412            4.,  5.,
413            6.,  7.,  8.,  9., 10., 11., 12., 11., 10.,  9.,  8.,
414            7.,  6.,
415            5.,  4.,  3.,  2.,  3.,  4.,  3.,  4.,  5.,  6.,  7.,
416            8.,  9.,
417            10., 11.])]
418     ]
419     [325.0 55.37994285714285
420     array([ 0.,  1.,  2.,  3.,  4.,  5.,  6.,  7.,  8.,  9., 10.,
421            11., 12.,
422            11., 10.,  9.,  8.,  7.,  6.,  5.,  4.,  3.,  2.,  3.,
423            4.,  5.,
424            6.,  7.,  8.,  9., 10., 11., 12., 11., 10.,  9.,  8.,
425            7.,  6.,
426            5.,  4.,  3.,  2.,  3.,  4.,  3.,  4.,  5.,  6.,  7.,
427            8.,  9.,
428            10., 11.])]
429     ]
430     [330.0 55.38397142857143
431     array([ 0.,  1.,  2.,  3.,  4.,  5.,  6.,  7.,  8.,  9., 10.,
432            11., 12.,
433            11., 10.,  9.,  8.,  7.,  6.,  5.,  4.,  3.,  2.,  3.,
434            4.,  5.,
435            6.,  7.,  8.,  9., 10., 11., 12., 11., 10.,  9.,  8.,
436            7.,  6.,
437            5.,  4.,  3.,  2.,  3.,  4.,  3.,  4.,  5.,  6.,  7.,
438            8.,  9.,
439            10., 11.])]
440     ]
441     [335.0 55.38799999999999
442     array([ 0.,  1.,  2.,  3.,  4.,  5.,  6.,  7.,  8.,  9., 10.,
443            11., 12.,
444            11., 10.,  9.,  8.,  7.,  6.,  5.,  4.,  3.,  2.,  3.,
445            4.,  5.,
446            6.,  7.,  8.,  9., 10., 11., 12., 11., 10.,  9.,  8.,
447            7.,  6.,
448            5.,  4.,  3.,  2.,  3.,  4.,  3.,  4.,  5.,  6.,  7.,
449            8.,  9.,
450            10., 11.])]
451     ]
452     [340.0 55.39202857142857
453     array([ 0.,  1.,  2.,  3.,  4.,  5.,  6.,  7.,  8.,  9., 10.,
454            11., 12.,
455            11., 10.,  9.,  8.,  7.,  6.,  5.,  4.,  3.,  2.,  3.,
456            4.,  5.,
457            6.,  7.,  8.,  9., 10., 11., 12., 11., 10.,  9.,  8.,
458            7.,  6.,
459            5.,  4.,  3.,  2.,  3.,  4.,  3.,  4.,  5.,  6.,  7.,
460            8.,  9.,
461            10., 11.])]
462     ]
463     [345.0 55.39605714285714
464     array([ 0.,  1.,  2.,  3.,  4.,  5.,  6.,  7.,  8.,  9., 10.,
465            11., 12.,
466            11., 10.,  9.,  8.,  7.,  6.,  5.,  4.,  3.,  2.,  3.,
467            4.,  5.,
468            6.,  7.,  8.,  9., 10., 11., 12., 11., 10.,  9.,  8.,
469            7.,  6.,
470            5.,  4.,  3.,  2.,  3.,  4.,  3.,  4.,  5.,  6.,  7.,
471            8.,  9.,
472            10., 11.])]
473     ]
474     [350.0 55.39999999999999
475     array([ 0.,  1.,  2.,  3.,  4.,  5.,  6.,  7.,  8.,  9., 10.,
476            11., 12.,
477            11., 10.,  9.,  8.,  7.,  6.,  5.,  4.,  3.,  2.,  3.,
478            4.,  5.,
479            6.,  7.,  8.,  9., 10., 11., 12., 11., 10.,  9.,  8.,
480            7.,  6.,
481            5.,  4.,  3.,  2.,  3.,  4.,  3.,  4.,  5.,  6.,  7.,
482            8.,  9.,
483            10., 11.])]
484     ]
485     [355.0 55.40402857142857
486     array([ 0.,  1.,  2.,  3.,  4.,  5.,  6.,  7.,  8.,  9., 10.,
487            11., 12.,
488            11., 10.,  9.,  8.,  7.,  6.,  5.,  4.,  3.,  2.,  3.,
489            4.,  5.,
490            6.,  7.,  8.,  9., 10., 11., 12., 11., 10.,  9.,  8.,
491            7.,  6.,
492            5.,  4.,  3.,  2.,  3.,  4.,  3.,  4.,  5.,  6.,  7.,
493            8.,  9.,
494            10., 1
```

```

361         11., 10., 9., 8., 7., 6., 5., 4., 3., 2., 3.,
        4., 5.,
362         6., 7., 8., 9., 10., 11., 12., 11., 10., 9., 8.,
        7., 6.,
363         5., 4., 3., 2., 3., 4., 3., 4., 5., 6., 7.,
        8., 9.,
364         10., 11.])
        ]
365 [310.0 55.361716666666666
366     array([ 0., 1., 2., 3., 4., 5., 6., 7., 8., 9., 10.,
367            11., 12.,
368            11., 10., 9., 8., 7., 6., 5., 4., 3., 2., 3.,
369            4., 5.,
370            6., 7., 8., 9., 10., 11., 12., 11., 10., 9., 8.,
371            7., 6.,
372            5., 4., 3., 2., 3., 4., 3., 4., 5., 6., 7.,
373            8., 9.,
374            10., 11.])
        ]
375 [315.0 55.354066666666666
376     array([ 0., 1., 2., 3., 4., 5., 6., 7., 8., 9., 10.,
377            11., 12.,
378            11., 10., 9., 8., 7., 6., 5., 4., 3., 2., 3.,
379            4., 5.,
380            6., 7., 8., 9., 10., 11., 12., 11., 10., 9., 8.,
381            7., 6.,
382            5., 4., 3., 2., 3., 4., 3., 4., 5., 6., 7.,
383            8., 9.,

```

```

382         10., 11.])
383     ]
384     [325.0 55.362275
385     array([ 0.,  1.,  2.,  3.,  4.,  5.,  6.,  7.,  8.,  9., 10.,
386            11., 12.,
387            11., 10.,  9.,  8.,  7.,  6.,  5.,  4.,  3.,  2.,  3.,
388            4.,  5.,
389            6.,  7.,  8.,  9., 10., 11., 12., 11., 10.,  9.,  8.,
390            7.,  6.,
391            5.,  4.,  3.,  2.,  3.,  4.,  3.,  4.,  5.,  6.,  7.,
392            8.,  9.,
393            10., 11.])
394     ]
395     [330.0 55.358725
396     array([ 0.,  1.,  2.,  3.,  4.,  5.,  6.,  7.,  8.,  9., 10.,
397            11., 12.,
398            11., 10.,  9.,  8.,  7.,  6.,  5.,  4.,  3.,  2.,  3.,
399            4.,  5.,
400            6.,  7.,  8.,  9., 10., 11., 12., 11., 10.,  9.,  8.,
401            7.,  6.,
402            5.,  4.,  3.,  2.,  3.,  4.,  3.,  4.,  5.,  6.,  7.,
403            8.,  9.,
404            10., 11.])
405     ]
406     [335.0 55.36412
407     array([ 0.,  1.,  2.,  3.,  4.,  5.,  6.,  7.,  8.,  9., 10.,
408            11., 12.,
409            11., 10.,  9.,  8.,  7.,  6.,  5.,  4.,  3.,  2.,  3.,
410            4.,  5.,
411            6.,  7.,  8.,  9., 10., 11., 12., 11., 10.,  9.,  8.,
412            7.,  6.,
413            5.,  4.,  3.,  2.,  3.,  4.,  3.,  4.,  5.,  6.,  7.,
414            8.,  9.,
415            10., 11.])
416     ]
417     [340.0 55.358416666666667
418     array([ 0.,  1.,  2.,  3.,  4.,  5.,  6.,  7.,  8.,  9., 10.,
419            11., 12.,

```



```
403         11., 10., 9., 8., 7., 6., 5., 4., 3., 2., 3.,
404         4., 5.,
405         6., 7., 8., 9., 10., 11., 12., 11., 10., 9., 8.,
406         7., 6.,
407         5., 4., 3., 2., 3., 4., 3., 4., 5., 6., 7.,
408         8., 9.,
409         10., 11.])
410     ]
411     [345.0 55.35536
412     array([ 0., 1., 2., 3., 4., 5., 6., 7., 8., 9., 10.,
413            11., 12.,
414            11., 10., 9., 8., 7., 6., 5., 4., 3., 2., 3.,
415            4., 5.,
416            6., 7., 8., 9., 10., 11., 12., 11., 10., 9., 8.,
417            7., 6.,
418            5., 4., 3., 2., 3., 4., 3., 4., 5., 6., 7.,
419            8., 9.,
420            10., 11.])
421     ]
422     [350.0 55.38452857142857
423     array([ 0., 1., 2., 3., 4., 5., 6., 7., 8., 9., 10.,
424            11., 12.,
425            11., 10., 9., 8., 7., 6., 5., 4., 3., 2., 3.,
426            4., 5.,
427            6., 7., 8., 9., 10., 11., 12., 11., 10., 9., 8.,
428            7., 6.,
429            5., 4., 3., 2., 3., 4., 3., 4., 5., 6., 7.,
430            8., 9.,
431            10., 11.])
432     ]
433     [355.0 55.3576875
434     array([ 0., 1., 2., 3., 4., 5., 6., 7., 8., 9., 10.,
435            11., 12.,
436            11., 10., 9., 8., 7., 6., 5., 4., 3., 2., 3.,
437            4., 5.,
438            6., 7., 8., 9., 10., 11., 12., 11., 10., 9., 8.,
439            7., 6.,
440            5., 4., 3., 2., 3., 4., 3., 4., 5., 6., 7.,
441            8., 9.,
442            10., 11.])
443     ]
```

```

424         10., 11.])
425     ]
426     [360.0 55.356337500000001
427     array([ 0.,  1.,  2.,  3.,  4.,  5.,  6.,  7.,  8.,  9., 10.,
428            11., 12.,
429            11., 10.,  9.,  8.,  7.,  6.,  5.,  4.,  3.,  2.,  3.,
430            4.,  5.,
431            6.,  7.,  8.,  9., 10., 11., 12., 11., 10.,  9.,  8.,
432            7.,  6.,
433            5.,  4.,  3.,  2.,  3.,  4.,  3.,  4.,  5.,  6.,  7.,
434            8.,  9.,
435            10., 11.])
436     ]
437     [365.0 55.358062499999996
438     array([ 0.,  1.,  2.,  3.,  4.,  5.,  6.,  7.,  8.,  9., 10.,
439            11., 12.,
440            11., 10.,  9.,  8.,  7.,  6.,  5.,  4.,  3.,  2.,  3.,
441            4.,  5.,
442            6.,  7.,  8.,  9., 10., 11., 12., 11., 10.,  9.,  8.,
443            7.,  6.,
444            5.,  4.,  3.,  2.,  3.,  4.,  3.,  4.,  5.,  6.,  7.,
445            8.,  9.,
446            10., 11.])
447     ]
448     [370.0 55.37388571428571
449     array([ 0.,  1.,  2.,  3.,  4.,  5.,  6.,  7.,  8.,  9., 10.,
450            11., 12.,
451            11., 10.,  9.,  8.,  7.,  6.,  5.,  4.,  3.,  2.,  3.,
452            4.,  5.,
453            6.,  7.,  8.,  9., 10., 11., 12., 11., 10.,  9.,  8.,
454            7.,  6.,
455            5.,  4.,  3.,  2.,  3.,  4.,  3.,  4.,  5.,  6.,  7.,
456            8.,  9.,
457            10., 11.])
458     ]
459     [375.0 55.35752
460     array([ 0.,  1.,  2.,  3.,  4.,  5.,  6.,  7.,  8.,  9., 10.,
461            11., 12.,
462            11., 10.,  9.,  8.,  7.,  6.,  5.,  4.,  3.,  2.,  3.,
463            4.,  5.,
464            6.,  7.,  8.,  9., 10., 11., 12., 11., 10.,  9.,  8.,
465            7.,  6.,
466            5.,  4.,  3.,  2.,  3.,  4.,  3.,  4.,  5.,  6.,  7.,
467            8.,  9.,
468            10., 11.])
469     ]
470     [380.0 55.35352
471     array([ 0.,  1.,  2.,  3.,  4.,  5.,  6.,  7.,  8.,  9., 10.,
472            11., 12.,
473            11., 10.,  9.,  8.,  7.,  6.,  5.,  4.,  3.,  2.,  3.,
474            4.,  5.,
475            6.,  7.,  8.,  9., 10., 11., 12., 11., 10.,  9.,  8.,
476            7.,  6.,
477            5.,  4.,  3.,  2.,  3.,  4.,  3.,  4.,  5.,  6.,  7.,
478            8.,  9.,
479            10., 11.])
480     ]
481     [385.0 55.35052
482     array([ 0.,  1.,  2.,  3.,  4.,  5.,  6.,  7.,  8.,  9., 10.,
483            11., 12.,
484            11., 10.,  9.,  8.,  7.,  6.,  5.,  4.,  3.,  2.,  3.,
485            4.,  5.,
486            6.,  7.,  8.,  9., 10., 11., 12., 11., 10.,  9.,  8.,
487            7.,  6.,
488            5.,  4.,  3.,  2.,  3.,  4.,  3.,  4.,  5.,  6.,  7.,
489            8.,  9.,
490            10., 11.])
491     ]
492     [390.0 55.34752
493     array([ 0.,  1.,  2.,  3.,  4.,  5.,  6.,  7.,  8.,  9., 10.,
494            11., 12.,
495            11., 10.,  9.,  8.,  7.,  6.,  5.,  4.,  3.,  2.,  3.,
496            4.,  5.,
497            6.,  7.,  8.,  9., 10., 11., 12., 11., 10.,  9.,  8.,
498            7.,  6.,
499            5.,  4.,  3.,  2.,  3.,  4.,  3.,  4.,  5.,  6.,  7.,
500            8.,  9.,
501            10., 11.])
502     ]
503     [395.0 55.34452
504     array([ 0.,  1.,  2.,  3.,  4.,  5.,  6.,  7.,  8.,  9., 10.,
505            11., 12.,
506            11., 10.,  9.,  8.,  7.,  6.,  5.,  4.,  3.,  2.,  3.,
507            4.,  5.,
508            6.,  7.,  8.,  9., 10., 11., 12., 11., 10.,  9.,  8.,
509            7.,  6.,
510            5.,  4.,  3.,  2.,  3.,  4.,  3.,  4.,  5.,  6.,  7.,
511            8.,  9.,
512            10., 11.])
513     ]
514     [400.0 55.34152
515     array([ 0.,  1.,  2.,  3.,  4.,  5.,  6.,  7.,  8.,  9., 10.,
516            11., 12.,
517            11., 10.,  9.,  8.,  7.,  6.,  5.,  4.,  3.,  2.,  3.,
518            4.,  5.,
519            6.,  7.,  8.,  9., 10., 11., 12., 11., 10.,  9.,  8.,
520            7.,  6.,
521            5.,  4.,  3.,  2.,  3.,  4.,  3.,  4.,  5.,  6.,  7.,
522            8.,  9.,
523            10., 11.])
524     ]
525     [405.0 55.33852
526     array([ 0.,  1.,  2.,  3.,  4.,  5.,  6.,  7.,  8.,  9., 10.,
527            11., 12.,
528            11., 10.,  9.,  8.,  7.,  6.,  5.,  4.,  3.,  2.,  3.,
529            4.,  5.,
530            6.,  7.,  8.,  9., 10., 11., 12., 11., 10.,  9.,  8.,
531            7.,  6.,
532            5.,  4.,  3.,  2.,  3.,  4.,  3.,  4.,  5.,  6.,  7.,
533            8.,  9.,
534            10., 11.])
535     ]
536     [410.0 55.33552
537     array([ 0.,  1.,  2.,  3.,  4.,  5.,  6.,  7.,  8.,  9., 10.,
538            11., 12.,
539            11., 10.,  9.,  8.,  7.,  6.,  5.,  4.,  3.,  2.,  3.,
540            4.,  5.,
541            6.,  7.,  8.,  9., 10., 11., 12., 11., 10.,  9.,  8.,
542            7.,  6.,
543            5.,  4.,  3.,  2.,  3.,  4.,  3.,  4.,  5.,  6.,  7.,
544            8.,  9.,
545            10., 11.])
546     ]
547     [415.0 55.33252
548     array([ 0.,  1.,  2.,  3.,  4.,  5.,  6.,  7.,  8.,  9., 10.,
549            11., 12.,
550            11., 10.,  9.,  8.,  7.,  6.,  5.,  4.,  3.,  2.,  3.,
551            4.,  5.,
552            6.,  7.,  8.,  9., 10., 11., 12., 11., 10.,  9.,  8.,
553            7.,  6.,
554            5.,  4.,  3.,  2.,  3.,  4.,  3.,  4.,  5.,  6.,  7.,
555            8.,  9.,
556            10., 11.])
557     ]
558     [420.0 55.32952
559     array([ 0.,  1.,  2.,  3.,  4.,  5.,  6.,  7.,  8.,  9., 10.,
560            11., 12.,
561            11., 10.,  9.,  8.,  7.,  6.,  5.,  4.,  3.,  2.,  3.,
562            4.,  5.,
563            6.,  7.,  8.,  9., 10., 11., 12., 11., 10.,  9.,  8.,
564            7.,  6.,
565            5.,  4.,  3.,  2.,  3.,  4.,  3.,  4.,  5.,  6.,  7.,
566            8.,  9.,
567            10., 11.])
568     ]
569     [425.0 55.32652
570     array([ 0.,  1.,  2.,  3.,  4.,  5.,  6.,  7.,  8.,  9., 10.,
571            11., 12.,
572            11., 10.,  9.,  8.,  7.,  6.,  5.,  4.,  3.,  2.,  3.,
573            4.,  5.,
574            6.,  7.,  8.,  9., 10., 11., 12., 11., 10.,  9.,  8.,
575            7.,  6.,
576            5.,  4.,  3.,  2.,  3.,  4.,  3.,  4.,  5.,  6.,  7.,
577            8.,  9.,
578            10., 11.])
579     ]
580     [430.0 55.32352
581     array([ 0.,  1.,  2.,  3.,  4.,  5.,  6.,  7.,  8.,  9., 10
```

```

445         11., 10., 9., 8., 7., 6., 5., 4., 3., 2., 3.,
        4., 5.,
446         6., 7., 8., 9., 10., 11., 12., 11., 10., 9., 8.,
        7., 6.,
447         5., 4., 3., 2., 3., 4., 3., 4., 5., 6., 7.,
        8., 9.,
448         10., 11.])
        ]
449 [380.0 55.3566125
450  array([ 0.,  1.,  2.,  3.,  4.,  5.,  6.,  7.,  8.,  9., 10.,
451 11., 12.,
452         11., 10., 9., 8., 7., 6., 5., 4., 3., 2., 3.,
        4., 5.,
453         6., 7., 8., 9., 10., 11., 12., 11., 10., 9., 8.,
        7., 6.,
454         5., 4., 3., 2., 3., 4., 3., 4., 5., 6., 7.,
        8., 9.,
455         10., 11.])
        ]
456 [385.0 55.361550000000001
457  array([ 0.,  1.,  2.,  3.,  4.,  5.,  6.,  7.,  8.,  9., 10.,
458 11., 12.,
459         11., 10., 9., 8., 7., 6., 5., 4., 3., 2., 3.,
        4., 5.,
460         6., 7., 8., 9., 10., 11., 12., 11., 10., 9., 8.,
        7., 6.,
461         5., 4., 3., 2., 3., 4., 3., 4., 5., 6., 7.,
        8., 9.,
462         10., 11.])
        ]
463 [390.0 55.357662499999996
464  array([ 0.,  1.,  2.,  3.,  4.,  5.,  6.,  7.,  8.,  9., 10.,
465 11., 12.,
        11., 10., 9., 8., 7., 6., 5., 4., 3., 2., 3.,
        4., 5.,
        6., 7., 8., 9., 10., 11., 12., 11., 10., 9., 8.,
        7., 6.,
        5., 4., 3., 2., 3., 4., 3., 4., 5., 6., 7.,
        8., 9.,

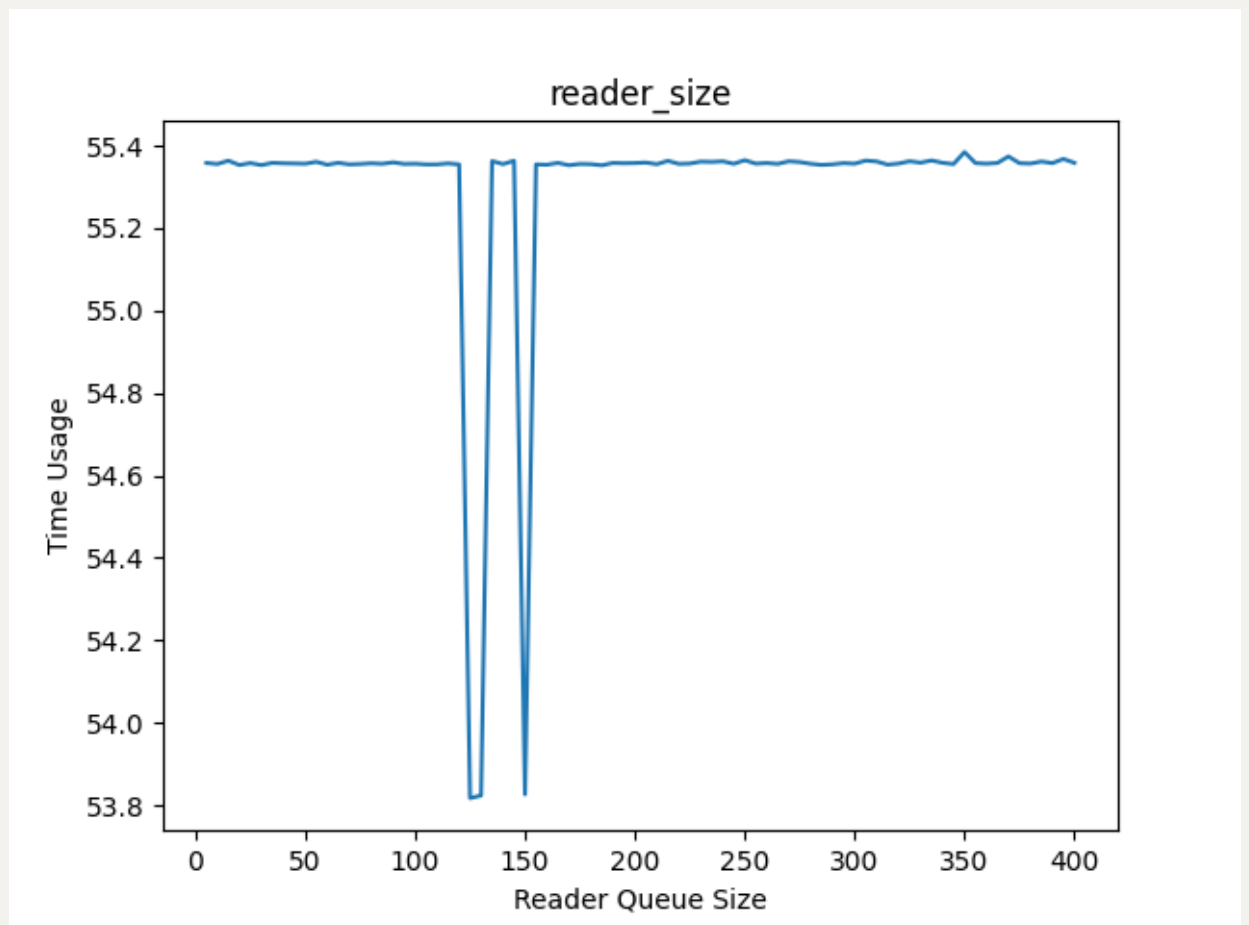
```

```

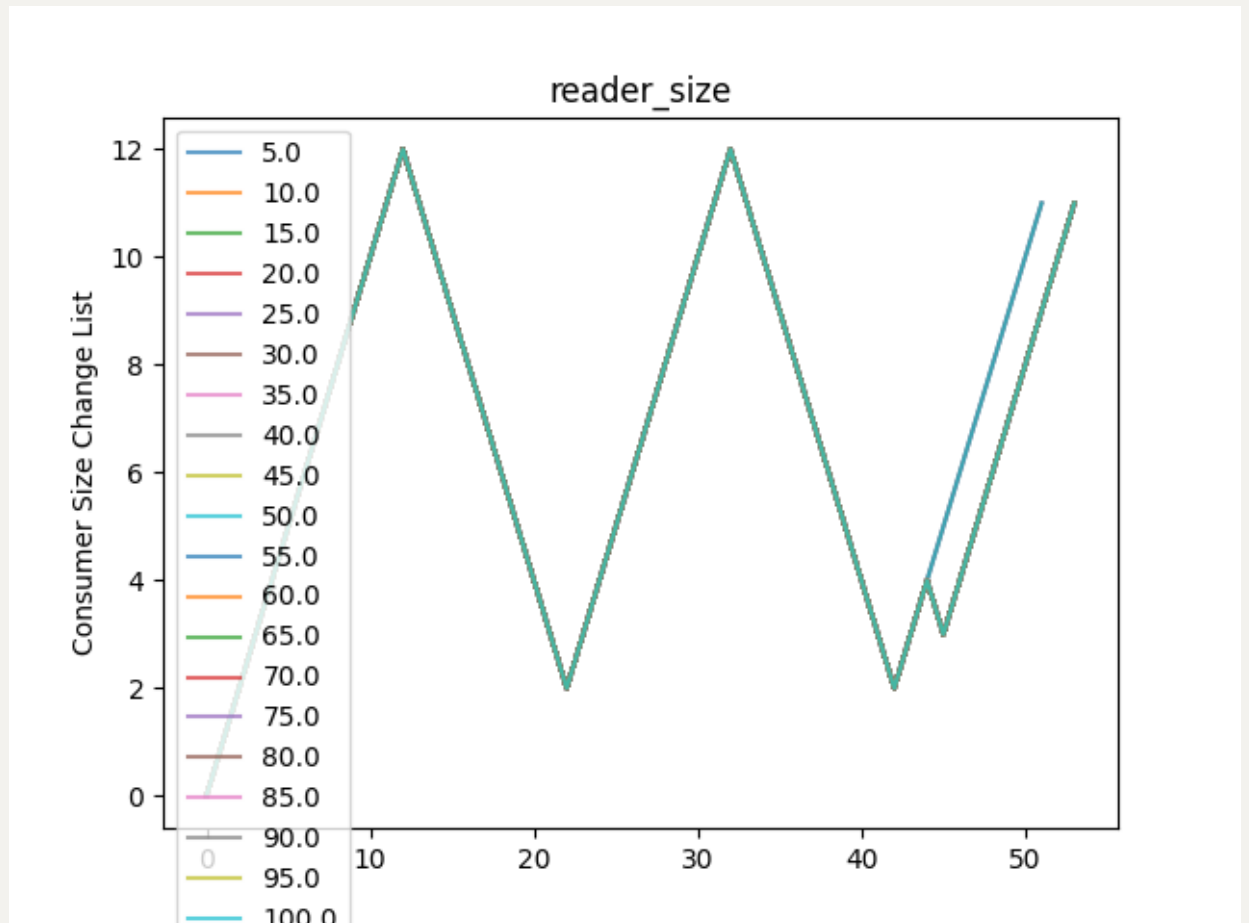
466         10., 11.])
467     ]
468     [395.0 55.36783333333333
469     array([ 0.,  1.,  2.,  3.,  4.,  5.,  6.,  7.,  8.,  9., 10.,
11., 12.,
470         11., 10.,  9.,  8.,  7.,  6.,  5.,  4.,  3.,  2.,  3.,
4.,  5.,
471         6.,  7.,  8.,  9., 10., 11., 12., 11., 10.,  9.,  8.,
7.,  6.,
472         5.,  4.,  3.,  2.,  3.,  4.,  3.,  4.,  5.,  6.,  7.,
8.,  9.,
473         10., 11.])
474     ]
475     [400.0 55.358314285714286
476     array([ 0.,  1.,  2.,  3.,  4.,  5.,  6.,  7.,  8.,  9., 10.,
11., 12.,
477         11., 10.,  9.,  8.,  7.,  6.,  5.,  4.,  3.,  2.,  3.,
4.,  5.,
478         6.,  7.,  8.,  9., 10., 11., 12., 11., 10.,  9.,  8.,
7.,  6.,
479         5.,  4.,  3.,  2.,  3.,  4.,  3.,  4.,  5.,  6.,  7.,
8.,  9.,
480         10., 11.])
481     ]

```

1. 圖像化 READER_QUEUE_SIZE 對 Time Usage 的影響:



2. 圖像化 READER_QUEUE_SIZE 對 CONSUMER SIZE CHANGE LIST 的影響:



3. 討論:

- a. 從圖中可以很明顯發現 `READER_QUEUE_SIZE` 約在 130 與 150 時, `Time Usage` 有明顯的下降, 其他則呈現平穩的狀況
- b. 從圖可以發現 `READER_QUEUE_SIZE` 對 `CONSUMER SIZE CHANGE` 並沒有太大影響, 幾乎所有的線重疊在一起, 總體波形呈現谷峰谷峰谷峰, 有條線岔出可以聯想到和第一點有些許點有極低值有相關
- c. What happens if `READER_QUEUE_SIZE` is very small?
 - i. 從這裡的結果來看, 對 `Time Usage` 與 `CONSUMER SIZE CHANGE` 並沒有太大影響

Feedback

楊子慶's feedback

本次的實作加入實驗的成分, 我利用 `Process` 的獨立性來測試當中多個 `Thread` 一起運行的表現, 以此實作獨立的運行環境, 讓使用者可以輕鬆調整操作變因來進行實驗。另運用 `Linux` 的 `Timer API` 實現 `Consumer` 數量的增減邏輯, 其運作方式頗為有趣, 是很好的練習經驗。

俞政佑's feedback

很少在資工的課程上面進行 "實驗", 是一次新鮮的體驗, 但感覺實驗之間的結果差距沒有很大, 分析感覺無法得到太多資訊。