实验报告2

一个基于 MESI 协议的缓存一致性模拟器，初始代码支持单个请求的顺序处理，但无法处理多个核心在同一时刻的并发读写请求，且仲裁优先级固定为按处理器 ID 排序。为了满足需求，对代码进行了修改，实现了并发请求处理和可配置的优先级仲裁功能。

修改内容

1. 支持并发请求处理

```
while (std::getline(file, line)) {
    std::istringstream iss(line);
    std::string request_str;
    std::vector<Request> requests;

    while (std::getline(iss, request_str, ';')) {
        request_str.erase(0, request_str.find_first_not_of(" \t"));
        request_str.erase(request_str.find_last_not_of(" \t") + 1);
        if (request_str != "NULL") {
            Request request = parseRequest(request_str);
            requests.push_back(request);
        }
    }
    std::cout << "\n----------Cycle " << cycle++ << "----------\n" << std::endl;
    if (!requests.empty()) {
        bus.arbitrate(requests);
    }
}
```

- **修改**:
  - 更新 `main.cpp`，按行读取输入文件，每行表示一个时间步的并发请求（以分号 `;` 分隔）。
  - 使用 `std::istringstream` 解析每行，跳过 `NULL` 请求，将有效请求存储到 `std::vector<Request>`。
- **效果**：程序能够正确解析和分组同一时刻的请求，为后续仲裁处理奠定基础

2. 实现总线仲裁

```
void Bus::arbitrate(const std::vector<Request> &requests) {
    // 按优先级排序请求（优先级值越小，优先级越高）
    std::vector<Request> sorted_requests = requests;
    std::sort(sorted_requests.begin(), sorted_requests.end(),
              [this](const Request &a, const Request &b) {
                  return priorities[a.processor_id] <
priorities[b.processor_id];
              });

    for (const Request &request : sorted_requests) {
        Cache *cache = caches[request.processor_id];
        std::cout << "\nProcessing " << request.toString()
                  << " (Priority: " << priorities[request.processor_id] << ")"
<< std::endl;
```

```
        if (request.op == READ) {
            uint16_t read_data = 0;
            cache->access(request.address, request.op, 0, &read_data);
        } else {
            cache->access(request.address, request.op, request.write_data);
        }
        cache->print_state();
    }
}
```

- **修改**：
  - 在 `bus.hpp` 中添加 `arbitrate` 方法，接受 `std::vector<Request>`，用于处理一组并发请求。
  - 在 `bus.cpp` 中实现 `arbitrate`，通过排序请求（初始按处理器 ID）依次调用 `Cache::access`，并打印状态。
  - 更新 `main.cpp`，将每行的请求传递给 `bus.arbitrate`，而不是直接调用 `Cache::access`。
  - 添加 `<algorithm>`（用于 `std::sort`）和 `"request.hpp"`（用于 `Request` 类型）到 `bus.cpp`。
- **效果**：支持同一时刻多个核心的并发请求处理，请求按处理器 ID 顺序执行，维护 MESI 协议的缓存一致性。

3. 实现可配置优先级仲裁

```
void Bus::setPriorities(const std::vector<int> &new_priorities) {
    if (new_priorities.size() != caches.size()) {
        std::cerr << "Error: initial_priorities size does not match the number
of cores." << std::endl;
        exit(1);
    }
    priorities = new_priorities;
}
```

- **修改**：
  - 在 `bus.hpp` 中添加 `std::vector<int> priorities` 成员，存储每个处理器的优先级（索引为 `processor_id`，值越小优先级越高）。
  - 更新 `Bus` 构造函数，接受 `initial_priorities` 参数，默认值 `{0, 1, 2, 3}`。
  - 添加 `setPriorities` 方法，允许运行时动态修改优先级，检查输入大小以确保有效性。
  - 修改 `arbitrate`，使用 `priorities` 数组排序请求（通过 `std::sort` 和 lambda 表达式）。
- **效果**：仲裁优先级可配置，允许用户通过 `setPriorities` 动态调整处理顺序，增加灵活性，同时保持 MESI 协议的正确性。

功能实现

1. **并发请求处理**：
   - 程序能够解析输入文件中每行表示的并发请求（格式如 `<P0, write, 256, 100>;<P1, write, 128, 200>;NULL;NULL`）。
   - 同一时刻的请求被分组并通过 `Bus::arbitrate` 处理，确保模拟多核心系统的并发行为。
2. **总线仲裁**：

- `Bus::arbitrate` 方法协调多个核心的请求，按指定顺序处理，调用 `Cache::access` 和 `broadcast` 维护 MESI 协议的缓存一致性。
- 每个请求处理后打印对应缓存状态，便于调试和验证。

3. **可配置优先级**：
   - 引入优先级向量，允许用户在初始化或运行时设置每个处理器的优先级。
   - 提供 `setPriorities` 方法，支持动态调整优先级。优先级值越小，请求越先处理，支持任意整数优先级配置。

测试结果

Conlict.txt

```
----------Cycle 0----------


Processing P0 W 0x100 0x100 (Priority: 0)
Cache State (Processor 0):
Set 0:  [T:0x8 S:M D:0x00000100 L:1]    [INVALID]
Set 1:  [INVALID]                       [INVALID]
Set 2:  [INVALID]                       [INVALID]
Set 3:  [INVALID]                       [INVALID]
Set 4:  [INVALID]                       [INVALID]
Set 5:  [INVALID]                       [INVALID]
Set 6:  [INVALID]                       [INVALID]
Set 7:  [INVALID]                       [INVALID]

Processing P1 W 0x80 0x200 (Priority: 1)
Cache State (Processor 1):
Set 0:  [T:0x4 S:M D:0x00000200 L:1]    [INVALID]
Set 1:  [INVALID]                       [INVALID]
Set 2:  [INVALID]                       [INVALID]
Set 3:  [INVALID]                       [INVALID]
Set 4:  [INVALID]                       [INVALID]
Set 5:  [INVALID]                       [INVALID]
Set 6:  [INVALID]                       [INVALID]
Set 7:  [INVALID]                       [INVALID]

----------Cycle 1----------


Processing P2 W 0x20 0x300 (Priority: 2)
Cache State (Processor 2):
Set 0:  [T:0x1 S:M D:0x00000300 L:1]    [INVALID]
Set 1:  [INVALID]                       [INVALID]
Set 2:  [INVALID]                       [INVALID]
Set 3:  [INVALID]                       [INVALID]
Set 4:  [INVALID]                       [INVALID]
Set 5:  [INVALID]                       [INVALID]
Set 6:  [INVALID]                       [INVALID]
Set 7:  [INVALID]                       [INVALID]

Processing P3 W 0x80 0x400 (Priority: 3)
Cache State (Processor 3):
Set 0:  [T:0x4 S:M D:0x00000400 L:1]    [INVALID]
Set 1:  [INVALID]                       [INVALID]
Set 2:  [INVALID]                       [INVALID]
```

```
Set 3:  [INVALID]                       [INVALID]
Set 4:  [INVALID]                       [INVALID]
Set 5:  [INVALID]                       [INVALID]
Set 6:  [INVALID]                       [INVALID]
Set 7:  [INVALID]                       [INVALID]

----------Cycle 2----------


Processing P0 R 0x20 0xffff (Priority: 0)
Cache State (Processor 0):
Set 0:  [T:0x8 S:M D:0x00000100 L:1]    [T:0x1 S:S D:0x00000300 L:2]
Set 1:  [INVALID]                       [INVALID]
Set 2:  [INVALID]                       [INVALID]
Set 3:  [INVALID]                       [INVALID]
Set 4:  [INVALID]                       [INVALID]
Set 5:  [INVALID]                       [INVALID]
Set 6:  [INVALID]                       [INVALID]
Set 7:  [INVALID]                       [INVALID]

Processing P1 R 0x20 0xffff (Priority: 1)
Cache State (Processor 1):
Set 0:  [T:0x1 S:S D:0x00000300 L:2]    [INVALID]
Set 1:  [INVALID]                       [INVALID]
Set 2:  [INVALID]                       [INVALID]
Set 3:  [INVALID]                       [INVALID]
Set 4:  [INVALID]                       [INVALID]
Set 5:  [INVALID]                       [INVALID]
Set 6:  [INVALID]                       [INVALID]
Set 7:  [INVALID]                       [INVALID]

Processing P2 R 0x80 0xffff (Priority: 2)
Cache State (Processor 2):
Set 0:  [T:0x1 S:S D:0x00000300 L:1]    [T:0x4 S:S D:0x00000400 L:2]
Set 1:  [INVALID]                       [INVALID]
Set 2:  [INVALID]                       [INVALID]
Set 3:  [INVALID]                       [INVALID]
Set 4:  [INVALID]                       [INVALID]
Set 5:  [INVALID]                       [INVALID]
Set 6:  [INVALID]                       [INVALID]
Set 7:  [INVALID]                       [INVALID]

Processing P3 R 0x100 0xffff (Priority: 3)
Cache State (Processor 3):
Set 0:  [T:0x4 S:S D:0x00000400 L:1]    [T:0x8 S:S D:0x00000100 L:2]
Set 1:  [INVALID]                       [INVALID]
Set 2:  [INVALID]                       [INVALID]
Set 3:  [INVALID]                       [INVALID]
Set 4:  [INVALID]                       [INVALID]
Set 5:  [INVALID]                       [INVALID]
Set 6:  [INVALID]                       [INVALID]
Set 7:  [INVALID]                       [INVALID]

----------Cycle 3----------


Processing P1 W 0x90 0x500 (Priority: 1)
Cache State (Processor 1):
```

```
Set 0:  [T:0x1 S:S D:0x00000300 L:2]    [INVALID]
Set 1:  [INVALID]                       [INVALID]
Set 2:  [INVALID]                       [INVALID]
Set 3:  [INVALID]                       [INVALID]
Set 4:  [T:0x4 S:M D:0x00000500 L:3]    [INVALID]
Set 5:  [INVALID]                       [INVALID]
Set 6:  [INVALID]                       [INVALID]
Set 7:  [INVALID]                       [INVALID]

Processing P2 W 0x80 0x77 (Priority: 2)
Cache State (Processor 2):
Set 0:  [T:0x1 S:S D:0x00000300 L:1]    [T:0x4 S:M D:0x00000077 L:3]
Set 1:  [INVALID]                       [INVALID]
Set 2:  [INVALID]                       [INVALID]
Set 3:  [INVALID]                       [INVALID]
Set 4:  [INVALID]                       [INVALID]
Set 5:  [INVALID]                       [INVALID]
Set 6:  [INVALID]                       [INVALID]
Set 7:  [INVALID]                       [INVALID]

----------Cycle 4----------


Processing P0 R 0x90 0xffff (Priority: 0)
Cache State (Processor 0):
Set 0:  [T:0x8 S:S D:0x00000100 L:1]    [T:0x1 S:S D:0x00000300 L:2]
Set 1:  [INVALID]                       [INVALID]
Set 2:  [INVALID]                       [INVALID]
Set 3:  [INVALID]                       [INVALID]
Set 4:  [T:0x4 S:S D:0x00000500 L:3]    [INVALID]
Set 5:  [INVALID]                       [INVALID]
Set 6:  [INVALID]                       [INVALID]
Set 7:  [INVALID]                       [INVALID]

Processing P3 W 0x80 0x96 (Priority: 3)
Cache State (Processor 3):
Set 0:  [T:0x4 S:M D:0x00000096 L:3]    [T:0x8 S:S D:0x00000100 L:2]
Set 1:  [INVALID]                       [INVALID]
Set 2:  [INVALID]                       [INVALID]
Set 3:  [INVALID]                       [INVALID]
Set 4:  [INVALID]                       [INVALID]
Set 5:  [INVALID]                       [INVALID]
Set 6:  [INVALID]                       [INVALID]
Set 7:  [INVALID]                       [INVALID]
```