generate_memory_topology 函数首先创建并调用 flatview_init 初始化一个空的 FlatView，接着调用 render_memory_region 进行树状 MemoryRegion 的展开，其中涉及了 128 位整数的计算，这里不深入细节，可以将其作为一个普通整数来理解。

render_memory_region 是内存平坦化的核心函数，下面对其进行详细分析。render_memory_region 函数比较复杂，本质作用就是将一个 MemoryRegion 转化成若干个 FlatRange，然后插入到第一个参数 FlatView 的 FlatRange 成员中。render_memory_region 有 5 个参数：第一个 view 表示的是一个全局 FlatView，就是一个 AddressSpace 表示的 FlatView；第二个是需要展开的 MemoryRegion；第三个是 base 值，表示的是即将展开的 MemoryRegion 的开始位置；第四个参数 clip 表示一段虚拟机物理地址范围；第五个参数 readonly 表示 MemoryRegion 是否可读。render_memory_region 函数会递归调用自己，每次传递子 region 作为第二个参数，并且将展开的 FlatRange 放到第一个参数 FlatView 的 ranges 数组成员中，每次 render_memory_region 函数返回都表示一段 MemoyRegion 展开完毕，当最上层的 render_memory_region 函数返回时，整个 AddressSpace 的根 MemoryRegion 展开完毕。假设待展开的根 MemoryRegion 为图 5-10 中的 1，2～5 为其对应的子 MemoryRegion。

首先将 1 表示的 MemoryRegion 作为 render_memory_region 的第二个参数，然后递归调用自身展开 2～5 子 Region。其调用关系如图 5-11 所示，只有当最底层的 4，5 展开之后，第 3 个 render_memory_region 调用才能返回，同样，只有当 2，3 展开之后，第一个 render_memory_region 调用才能返回。

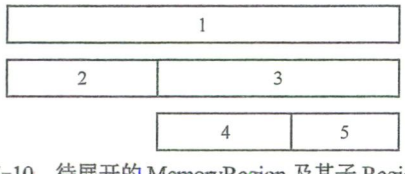

图 5-10　待展开的 MemoryRegion 及其子 Region

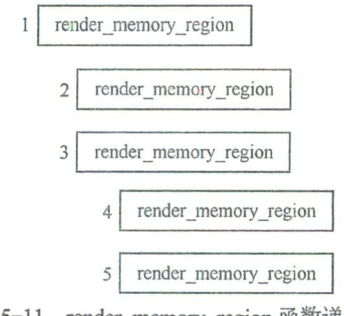

图 5-11　render_memory_region 函数递归调用

由于需要考虑到各种情况，render_memory_region 函数很复杂，这里通过举例来完成这个函数的讲解。图 5-12 展示了一个 mr 和虚拟机物理地址空间。mr 有一个子 MemoryRegion——mr1，虚拟机物理地址空间已经展开了两个 FlatRange，分别是 range1 和 range2。

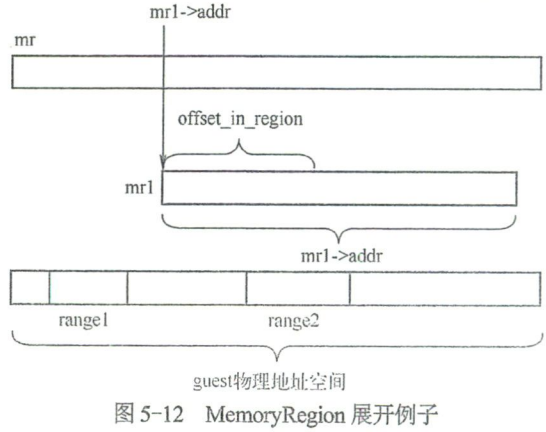

图 5-12　MemoryRegion 展开例子