# 4.3.2 精化链

本文将解析 ‘Refine’ 类型的处理链，并说明其如何通过改变输入的组织和输出的生成方式来提高文档搜索的质量。

#### ##4 1. 整体流程

‘Refine’ 文档处理链通过遍历输入文档并迭代更新其答案来构建响应。对于每个文档，它将所有非文档输入（例如用户的问题或其他与当前文档相关的信息）、当前文档和最新的中间答案传递给语言模型 (LLM)，以获得新的答案。

#### ##4 2. 遍历文档阶段

在这个阶段，系统会遍历输入的所有文档。对于每个文档，一起作为提示(Prompt)传递给 LLM 的内容有：

* 一些上下文信息，例如用户的问题或其他与当前文档相关的信息。
* 最新的中间答案。中间答案是系统在处理之前的文档时产生的。一开始，中间答案可能是空的，但随着系统处理更多的文档，中间答案会不断更新。
* 当前文档。

与Map reduce 链和重排链不同的是它不产生新文档，只不断更新的是提示，迭代出更全面的答案。而且文档之间的影响是传递性的，上一个文档形成的答案会影响下一个文档的答案。

#### ##4 3. 更新答案阶段

在这个阶段，系统将提示传递给 LLM，然后将 LLM 生成的答案作为新的中间答案。这个过程会迭代进行，直到所有的文档都被处理。

#### ##4 4. 最终实现效果

通过 ‘Refine’ 文档处理链，系统可以对包含多个文档的问题生成一个全面的答案，而且每个文档的处理结果都会影响后续文档的处理。这种处理方式可以提高文档搜索的质量，特别是在处理大量文档的情况下。

#### ##4 5. 适用场景

‘Refine’ 类型的处理链方式主要适用于处理大量文档的情况，特别是当这些文档不能全部放入模型的上下文中时。然而，这种处理方式可能会使用更多的计算资源，并且在处理某些复杂任务（如文档之间频繁地交叉引用，或者需要从许多文档中获取详细信息）时可能表现不佳。

总的来说，通过使用 ‘Refine’ 文档处理链，系统可以有效地处理大量文档的情况，从而提高文档搜索的质量。然而，这种处理方式可能需要更多的计算资源，并且可能在处理复杂任务时表现不佳。