# 合并文档链概述

在许多应用场景中，我们需要与文档进行交互，如阅读说明书、浏览产品手册等等。近来，基于这些场景开发的应用，如chatDOC和chatPDF，都受到了广大用户的欢迎。为了满足对特定文档进行问题回答、提取摘要等需求，Langchain设定了几种合并文档链类型。

这些核心链都是为处理文档而设计的。它们在对文档进行概括、回答文档问题、从文档中提取信息等方面非常有用。

但是文档链的类型给初学者造成了很大的困扰。主要是因为我们通常不清楚在指定了这些类型后，中间的处理流程发生了什么。如果我们能从各个类型的具体步骤进行理解，就会发现，这些类型的主要区别在于它们处理输入文档的方式，以及在中间过程中与模型的交互次数和答案来源于哪些阶段。理解了这些，我们就可以更清楚地认识到各种类型的优缺点，从而在生产环境中做出更好的决策。

换句话说，一旦我们理解了每个类型的具体步骤提交了什么提示(Prompt)，提示从何而来就可以明确知道使用哪种类型更符合我们的需求。我们会在后面对每个类型经历的具体步骤进行拆解。在这里我们先做个概述，没看懂可移步相应的文档类型的小节。

“Stuff链” 是处理文档链中最直接的一个。它接收一组文档，将它们全部插入到一个提示中，然后将该提示传递给LLM。这种链适合于文档较小且大部分调用只传入少量文档的应用。

“精化（Refine）”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、当前文档和最新的中间答案传递给LLM链，以获得新的答案。

由于精化链一次只向LLM传递一个文档，因此它非常适合需要分析比模型上下文能容纳更多的文档的任务。但显然，这种链会比如Stuff链这样的链调用更多的LLM。此外，还有一些任务很难通过迭代来完成。例如，当文档经常相互交叉引用或任务需要许多文档的详细信息时，精化链的表现可能较差。

“Map Reduce”首先将LLM链单独应用于每个文档（Map步骤），并将链输出视为新的文档。然后，它将所有新文档传递给一个单独的“Combine Documents Chain”，以获得单一的输出（Reduce步骤）。它可以选择首先压缩或合并映射的文档，以确保它们适合“Combine Documents Chain”（这将经常将它们传递给LLM）。如果需要，这个压缩步骤将递归地执行。

“重排链（Map Re-rank）”对每个文档运行初始提示，不仅试图完成任务，还对其答案的确定程度给出评分。得分最高的响应将被返回。