课题二前端设计

本篇文档主要讲述课题二对于算法前端展示的一些初步设想，旨在为课题五的方案实现提供可行性分析。

课题二的前端流程展示主要分成三步：首先展示算子库，让用户知悉目前有哪些算子可以使用，算子具体的功能和信息。接下来进入代码编辑阶段，这部分则按照 VSCode 已有的界面制作即可。 最后在用户点击编译之后，进入一个新的前端展示页面，利用编译器编译完所提供的信息，展示当前编写程序的任务划分和算子映射。 算子映射的部分之后也可以和课题三的核组库结合一起展示。

图 1　展示流程示意

算子库前端设计

算子库将会为前端提供固定的算子数据，需要做成一个带子集菜单的算子展示页面。

密码学部分目前主要有 md5, sm3, sha1 三个算子，因为密码学本身已经是算子了，所以只需要做二级菜单。用户点击算子对应按钮时展开一个新菜单，里面带有算子的功能介绍。

神经网络部分则是由神经网络，展开到算子菜单，再展示到功能介绍菜单，做三层的菜单。

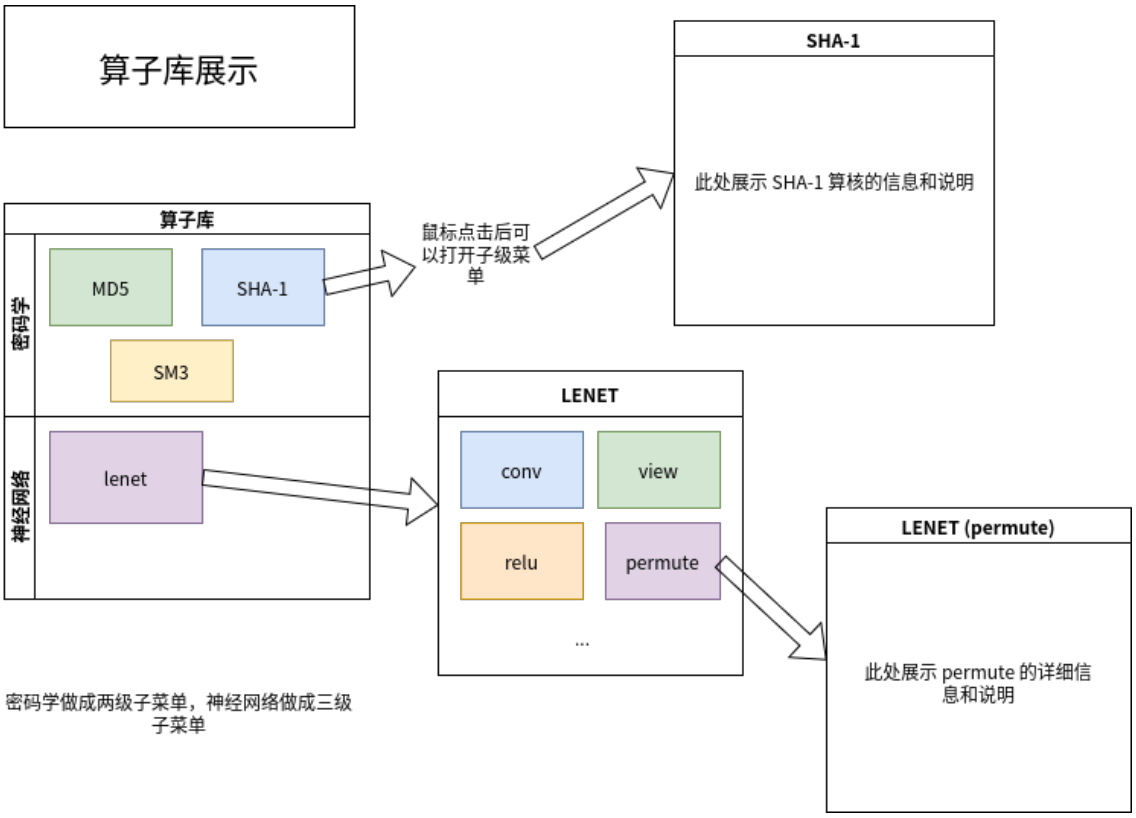

图 2　算子库示意

流程图前端设计

密码学部分的前端展示主要需要达成算法的分解、部署和映射等目标。为了更清晰地阐述这些目标，以下将按照这三个方面分别叙述课题二对前端部分的设想。

首先，用户可以在前端提供的编辑器上编写代码并选择编译方式。在代码编写完成后，用户可以通过一个简单易用的按钮，将 `args.json` 的参数以表单的形式呈现给用户进行选择。 前端将根据用户的选择重新序列化为 JSON 文件，以便提供给编译器进行后续处理。

用户点击“编译”按钮后，将进入展示页面：

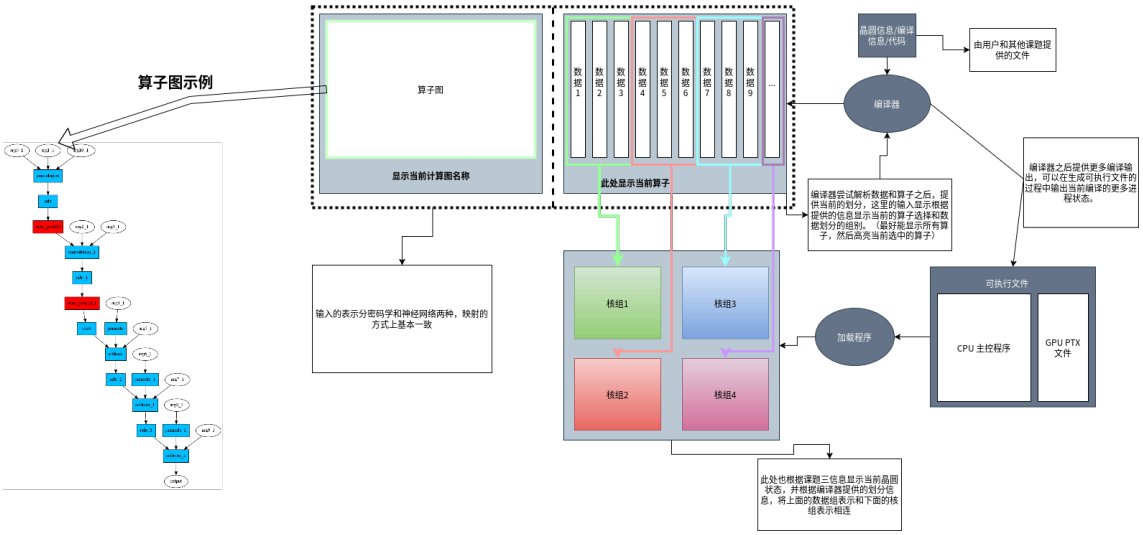

图 3　流程图示意

在展示页面中，首先需要提供一个编译执行输出的展示窗口。该窗口将显示编译器的输出信息，并在后续版本中加入更多状态信息，以便实时展示当前的编译进度和状态变化。

在密码学部分的优化实现中，我们主要关注如何将大量独立输入按照负载比例分成不同的任务组，并在不同的核组上并行运行。编译器将通过解析用户输入，将用户程序的一些关键信息进行提取。这些信息包括计算输入的大小、分组方式、选择的算子等，最终序列化成 JSON 文件以供前端使用。前端根据这些数据显示当前算子的详细信息以及数据组别划分情况。为了使得展示内容更加直观，可以将输入用一个大的矩形表示，而在矩形内部则用多个长条形表示不同的独立输入。整个大输入矩阵通过一条线连接到编译器，同时还需添加晶圆信息的输入，与表示可执行文件的矩形相连接。这一部分构成了第一部分的分解解析，同时也可以作为编译过程中的展示内容。

神经网络的展示和密码学类似，只是需要将对应的数据表现，换成算子的名称。

编译完成后，编译器还可以提供当前输出文件路径的信息，这些信息也应当在前端上进行展示，以便用户能够方便地获取相关结果和文件位置。可以用一个矩形代表可执行文件，分为两大部分：一部分是供 GPU 运行的 PTX 文件，另一部分则是供 CPU 运行的计算部分和主控部分。在前端展示中，可以将它们用一个矩形划为主体，再加上两个子矩形，分别代表不同的子部分，然后将编译器提供的路径信息填在上面，并与晶圆部分通过线条相连，以直观地表示算法的部署情况。

如上所述，在分解部分完成后，各输入会根据负载比例分别加载到不同的核组上。因此，对于映射部分，目前构思是在编译完成后，通过之前所提供的信息，将不同的数据分区块和算子分配到相应算核进行链接。在前端表示上，可以将前一部分的数据区块和算子连接到代表晶圆的大矩形上， 从而形成一个完整而清晰的数据流动图示，使得用户能够直观理解整个算法执行过程及其背后的逻辑关系。