2022 年中国研究生数学建模竞赛 D 题（中兴题目）

PISA 架构芯片资源排布问题

一、背景介绍

芯片是电子行业的基础，在当前日益复杂的国际形势下，芯片成了各个大国必争的高科技技术。本课题关注网络通信领域的交换芯片，传统的交换芯片功能固定，当出现新的网络协议时，必须重新设计芯片，而芯片从设计到使用，往往需要几年的时间，因此固定功能的交换芯片大大降低了研发效率，为了解决此问题，诞生了可编程的交换芯片。PISA（Protocol Independent Switch Architecture）是当前主流的可编程交换芯片架构之一，其有着和固定功能交换芯片相当的处理速率，同时兼具了可编程性，在未来网络中具有广阔的应用场景[1-2]。

在对 PISA 架构作进一步说明之前我们首先澄清几个基本概念：

1. 报文：报文即网络通信中传输的数据包，在网络通信中，用户传输的数据被封装成一个个的数据包进行传输。

2. 基本块：基本块是源程序的一个程序片段，对源程序进行基本块划分会将源程序划分为一个个的基本块。至于基本块如何划分本身也是一个值得探讨的问题，但超出了本问题的范围，在此不再多加赘述。

3. 流水线：流水线为一系列处理单元串联构成，报文在流水线中按次序依次通过每个处理单元，最终完成处理。流水线各级即是指流水线中各处理单元。

PISA 架构如图 1 所示，其包括报文解析（parser）、多级的报文处理流水线（Pipeline Pocket Process）、以及报文重组（Deparser）三个组成部分。报文解析用于识别报文种类；多级的报文处理流水线用于修改报文数据，在实际的 PISA 架构芯片中，不同的芯片流水线的级数可能不同；报文重组用于报文重新组装。本课题只关注其中多级的报文处理流水线部分。在 PISA 架构编程模型中，用户使用 P4 语言描述报文处理行为得到 P4 程序，再由编译器编译 P4 程序，进而生成芯片上可以执行的机器码。编译器在编译 P4 程序时，会首先将 P4 程序划分为一系列的基本块，再将各基本块排布到流水线各级当中。由于各基本块均会占用一定的芯片资源，将基本块排布到流水线各级即是将各基本块的资源排布到流水

线各级当中（即需要确定每个基本块排布到流水线哪一级），因此我们将基本块的排布问题称作 PISA 架构芯片资源排布问题。在实际的 PISA 架构芯片的设计中，为了减少连线的复杂度，往往对流水线各级的资源、以及流水线各级之间的资源有着多种多样的约束条件，这一系列复杂的资源约束条件使得资源排布问题尤为困难。然而，芯片的各类资源均有限，越高的资源利用率意味着能够越好的发挥芯片的能力，让芯片支持更多的业务，因此，高资源利用率的资源排布算法对于编译器设计至关重要。

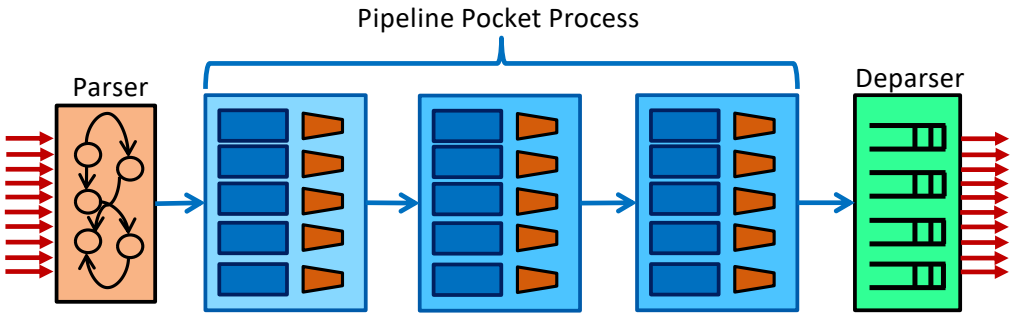

图 1　PISA 架构图

二、问题描述

由基本块定义可知基本块为源程序的片段，基本块中会执行指令来完成计算，指令执行时会读取指令源操作数（即读源操作数对应的变量）进行计算，再将计算结果赋值给目的操作数（即写目的操作数对应的变量）。对于划分好的基本块，每个基本块中的指令并行执行，执行时按照先读后写的顺序（由芯片底层实现所决定），即先同时读出所有的目的操作数，再并行执行所有指令的计算，最后同时将计算结果赋值给目的操作数。由于并行向同一个变量写多次时存在冲突，因此每个基本块只会写同一个变量一次（即基本块中不存在多条指令对相同变量赋值）。

基本块可以被抽象成一个节点，抽象后基本块中执行的具体指令被屏蔽，只保留读写的变量信息。当基本块 A 执行完可以跳转到基本块 B 执行时，在 A 和 B 之间增加一条有向边，这样 P4 程序即可表示为一个有向无环图（P4 程序不存在循环），称作 P4 程序流程图，如图 2 左图所示。PISA 架构资源排布即是将 P4 程序流程图中的各节点（即各基本块）在满足约束条件下排布到流水线各级当中。约束条件来自于两方面，一方面来自于 P4 程序本身，P4 程序每个基本块均会写一部分变量（即对变量赋值）和读一部分变量，变量的读写使得基本块之间存在

数据依赖，同时，基本块执行完后可能跳转到多个基本块执行，从而使得基本块之间也存在着控制依赖，数据依赖和控制依赖约束了基本块排布的流水线级数的大小关系，关于数据依赖和控制依赖的详细说明参见附录 A；另一方面的约束条件来自于芯片的资源约束，芯片中的资源包括 TCAM、HASH、ALU、QUALIFY 四类（附录 B 中解释了四类资源的作用以供感兴趣的同学进一步了解，不了解并不影响问题作答）。流水线中针对于这四类资源有着严格的限制（具体的资源限制在赛题中进行说明），资源排布时不能违反芯片的资源限制。

本问题中，输入数据给出了各基本块在 P4 程序流程图中的邻接关系，各基本块占用的四类资源的数量，以及各基本块读写的变量信息，本问题的赛题总共包括两个子问题，需要同学们在满足上述数据依赖、控制依赖、以及各具体子问题的资源约束条件下进行资源排布，并充分考虑各子问题的优化目标，以求最大化芯片资源利用率。

为了帮助大家进一步理解本问题，在附录 C 中给出了一个简单的资源排布示例。

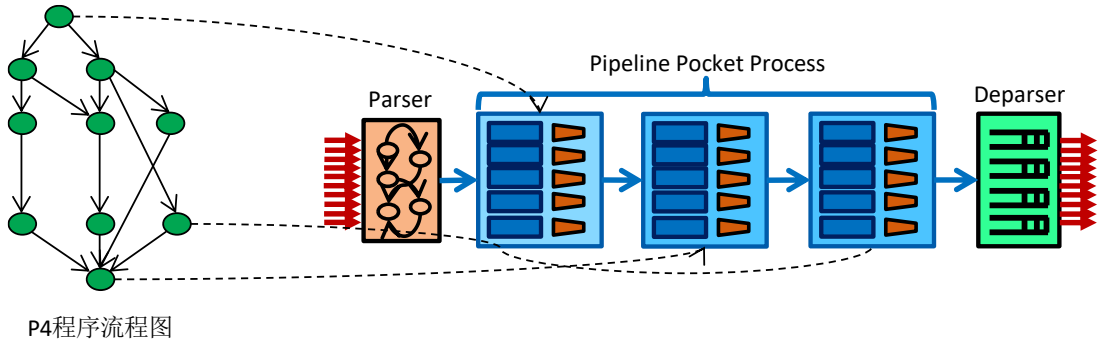

图 2　PISA 架构资源排布示意图

三、输入数据说明

输入数据包含三个附件，分别给出了各基本块资源使用情况、各基本块读写的变量信息、以及各基本块在流图中的邻接基本块，各附件格式如下：

（1）attachment1.csv：各基本块使用的资源信息

BLOCK	TCAM	HASH	ALU	QUALIFY
0	0	0	2	0

1	0	0	0	0
2	0	0	0	0
3	0	0	0	0
4	0	0	10	3
...	...	...	...	...

表中第一列为基本块编号，第 2 列到第 5 列为各基本块使用的四种资源的数目，资源总共分为四种（TCAM、HASH、ALU、QUALIFY），比如由上表可以知道，0 号基本块需要占用 2 个 ALU 资源，4 号基本块需要占用 10 个 ALU 资源和 3 个 Qualify 资源。

（2）attachment2.csv：各基本块读写的变量信息

0	W	X0	X1	...
0	R			...
1	W			...
1	R			...
2	W			...
2	R			...
3	W			...
3	R			...
4	W	X5	X6	...
4	R	X23	X24	...
...	...	...	...	...

表中第一列表示基本块编号，第二列中“W”表示写，“R”表示读，后续列表示本基本块写或者读的变量，当某一行从第三列及后续列没有任何元素时，说明此编号的基本块没有写（或读）任何变量（此时该基本块单纯作为连接其它基本块的中间基本块，没有执行任何计算）。例如：0 号基本块写了变量 X0、X1，但没有读任何变量，1 号基本块既没有写任何变量，也没有读任何变量。

（3）attachment3.csv：各基本块在流程图中的邻接基本块信息

0	1	2		...
1	2			...

2				...
3	31			...
4	0	586		...
...	...	...	...	...

表中第一列为基本块编号，后续列为与当前基本块在流程图中邻接的基本块编号，即在流程图（如图 2 左图所示）中，本基本块到后续列中的基本块之间存在有向边连接。比如，由上表第一行可知，0 号基本块到 1 号基本块和 2 号基本块之间存在有向边连接（即 0 号基本块执行结束后可以跳转到 1 号基本块或 2 号基本块执行），由第三行可知从 2 号基本块出发没有边（即基本块 2 执行后程序结束，不会再跳转到其它基本块执行）。通过此文件可以确定基本块在源程序的执行顺序，确定每个基本块执行后跳转的目的基本块，进而构建起基本块的流程图。

四、问题

本课题需要建立资源排布问题的数学模型，并在此基础上处理如下两个问题。

问题 1：给定资源约束条件如下：

（1）流水线每级的 TCAM 资源最大为 1；

（2）流水线每级的 HASH 资源最大为 2；

（3）流水线每级的 ALU 资源最大为 56；

（4）流水线每级的 QUALIFY 资源最大为 64；

（5）约定流水线第 0 级与第 16 级，第 1 级与第 17 级，...，第 15 级与第 31 级为折叠级数，折叠的两级 TCAM 资源加起来最大为 1，HASH 资源加起来最大为 3。注：如果需要的流水线级数超过 32 级，则从第 32 开始的级数不考虑折叠资源限制；

（6）有 TCAM 资源的偶数级数量不超过 5；

（7）每个基本块只能排布到一级。

在上述资源约束条件下进行资源排布，并以占用的流水线级数尽量短为优化目标。请给出资源排布算法，输出基本块排布结果，输出的结果格式如下：

流水线级数	分配的基本块编号			
0	xxx	xxx	xxx	xxx
1	xxx	xxx	xxx	xxx
...	...	...	...	...
N	xxx	xxx	xxx	xxx

问题 2：考虑如图 3 所示的流图，基本块 2 和基本块 3 不在一条执行流程上（因为基本块 1 执行完后要么执行基本块 2，要么执行基本块 3，基本块 2 和基本块 3 不可能都执行）。准确来说，在 P4 程序流程图中，由一个基本块出发可以到达另一个基本块则两基本块在一条执行流程上，反之不在一条执行流程上。对于这种不在一条执行流程上的基本块，可以共享 HASH 资源和 ALU 资源，基本块 2 和 3 中任意一个的 HASH 资源与 ALU 资源均不超过每级资源限制，基本块 2 和 3 即可排布到同一级。据此，对问题 1 中的约束条件（2）、（3）、（5）作如下更改：

（2）流水线每级中同一条执行流程上的基本块的 HASH 资源之和最大为 2；

（3）流水线每级中同一条执行流程上的基本块的 ALU 资源之和最大为 56；

（5）折叠的两级，对于 TCAM 资源约束不变，对于 HASH 资源，每级分别计算同一条执行流程上的基本块占用的 HASH 资源，再将两级的计算结果相加，结果不超过 3。

其它约束条件同问题 1，更改资源约束条件后重新考虑问题 1，给出排布算法，输出基本块排布结果。输出格式同问题 1。

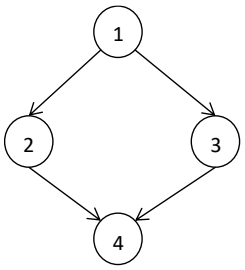

图 3　流图示例

参考文献

1. Bosshart P, Daly D, Gibb G, et al. P4: Programming protocol-independent packet processors[J]. ACM SIGCOMM Computer Communication Review, 2014, 44(3): 87-95.
2. Bosshart P, Gibb G, Kim H S, et al. Forwarding metamorphosis: Fast programmable match-action processing in hardware for SDN[J]. ACM SIGCOMM Computer Communication Review, 2013, 43(4): 99-110.

附录 A

1、数据依赖

数据依赖是语句或代码块间数据流造成的一种约束。具体来说包含 3 种依赖方式：

（1）S1 在 S2 之前执行，当 S1 写某个变量而 S2 读该变量时，S1 和 S2 存在写后读数据依赖。例如图 A-1 中，S1 写了变量 a，而 S2 读了变量 a（使用 a 作条件判断），则 S1 和 S2 存在写后读依赖；S3 写了变量 d，而 S4 读了变量 d（使用 d 作为指令的源），则 S3 和 S4 也存在写后读依赖。

（2）S1 在 S2 之前执行，当 S1 读某个变量而 S2 写该变量时，S1 和 S2 存在读后写数据依赖。例如图 A-1 中，S3 读了变量 e（使用 e 作指令的源），而 S4 写了变量 e，则 S3 和 S4 存在读后写依赖。

（3）S1 在 S2 之前执行，当 S1 和 S2 均写某个变量时，S1 和 S2 存在写后写数据依赖。例如图 A-1 中，S3 和 S5 均写了变量 d，则 S3 和 S5 存在写后写依赖。

```
S1          a = b + c
S2          if a > 10 goto L1
S3            d = b * e
S4            e = d + 1
S5  L1:     d = e/2
```

图 A-1　数据依赖示例

在 PISA 架构中，当基本块 A 和 B 存在写后读数据依赖或写后写数据依赖

时，基本块 A 排布的流水线级数需要小于基本块 B 排布的级数，比如基本块 B 被排到了流水线第 10 级，则基本块 A 只能排到流水线第 0 级到第 9 级；当基本块 A 和 B 存在读后写数据依赖时，基本块 A 排布的流水线级数需要小于或等于基本块 B 排布的级数，比如基本块 B 被排到流水线第 10 级，则基本块 A 只能排到第 0 级到第 10 级。

2、控制依赖

控制依赖是程序控制流导致的一种约束。控制依赖定义为：当从某个基本块出发的路径，只有部分路径通过下游某个基本块时，两基本块构成控制依赖。如图 A-2 所示的控制流图，B1 与 B2、B5、B6、B7、B8 之间存在控制依赖，B5 与 B6、B8 之间存在控制依赖，但 B5 和 B7 不构成控制依赖，因为从 B5 出发的两条路径下游都会经过 B7。在 PISA 架构中，如果基本块 A 与基本块 B 存在控制依赖，则 A 排布的流水线级数需要小于或等于 B 排布的流水线级数。

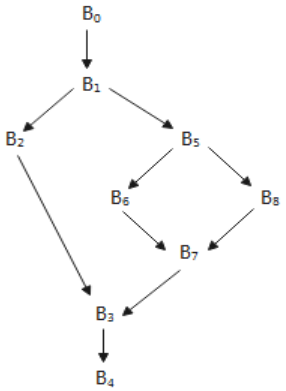

图 A-2　控制依赖示例

附录 B

为了说明 TCAM、HASH、ALU、QUALIFY 四类资源的作用，我们需要对 P4 语言有一个简单的了解。表项是 P4 程序的基本元素之一，P4 程序中的表项包括 key 和 action 两个基本属性，表项在芯片底层由一条条的条目构成，P4 程序执行表项时会使用 key 去条目中查找，命中一个条目后会从条目中取出源数据，再将取出的数据放到 action 的指令中去执行。关于表项的详细说明可参见如下网址的 P4 语言标准文档：https://p4.org/p4-spec/docs/P4-16-v1.0.0-spec.html.

在简单了解 P4 语言后，我们进一步说明四类资源的具体作用。

（1）TCAM 和 HASH：TCAM 和 HASH 都是芯片中的表项资源，P4 程序中可定义类型为 TCAM 或 HASH 的表项，类型为 TCAM 的表项需要占据芯片 TCAM 资源，类型为 HASH 的表项需要占用芯片 HASH 资源；

（2）ALU：如上所述，表项执行时需要从条目中取出源数据放到 action 的指令中执行，指令的执行在芯片中需要放到 ALU 中进行，因此 action 中的指令需要占用 ALU 资源；

（3）QUALIFY：在 P4 程序中有做条件判断的 if-else 语句（含义同其它高级语言），if-else 语句中的条件判断在执行时需要放到芯片 QUALIFY 中进行，需要占用 QUALIFY 资源。

附录 C

资源约束以赛题中问题 1 的资源约束为准，假设输入数据如下：

（1）attachment1.csv：各基本块使用的资源信息

BLOCK	TCAM	HASH	ALU	QUALIFY
0	0	0	2	0
1	1	0	0	0
2	0	0	0	0
3	1	0	0	0

（2）attachment2.csv：各基本块读写的变量信息

0	W	X0		
0	R			
1	W	X1		
1	R	X0		
2	W	X2		
2	R	X0		

（3）attachment3.csv：各基本块在流程图中的邻接基本块信息

0	1	2		...
1	3			...
2	3			...
3				...

依据 attachment3.csv，可以构建起基本块执行的流程图，如图 C-1 所示。依据基本块流程图和控制依赖的定义可以知道：基本块 0 和基本块 1 存在控制依赖，基本块 0 和基本块 2 也存在控制依赖。在流程图中，1 号基本块和 2 号基本块位于 0 号基本块下游，由于 0 号基本块写了变量 X0，1 号基本块和 2 号基本块均读了变量 X0，因此 0 号基本块和 1 号基本块存在写后读数据依赖，0 号基本块和 2 号基本块也存在数据依赖。综合依赖关系来看，1 号基本块和 2 号基本块排布的级数需要大于 0 号基本块排布的级数。

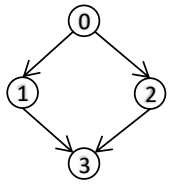

图 C-1　基本块流程图示例

在资源排布时，假设 0 号基本块排布到流水线第 0 级，则 1 号基本块和 2 号基本快排布的级数需要大于等于 1，由于 1 号基本块和 2 号基本块放到一级时满足资源约束，因此 1 号基本块和 2 号基本块可以同时放到第 1 级。3 号基本块放到第 1 级时超过资源约束（1 号基本块和 3 号基本块均需要占据 1 个 TCAM 资源，而资源约束中一级流水线 TCAM 资源最多为 1），因此 3 号基本块不能放到流水线第 1 级。将 3 号基本块放到第 0 级和第 2 级均可以满足依赖关系和资源约束，放到第 0 级时总共占用流水线 2 级，而放到第 2 级时总共占用流水线 3 级，因此将基本块 3 放到第 0 级更优。因此最优排布结果如下：

流水线级数	分配的基本块编号	
0	0	3
1	1	2

事实上，最优排布结果更改了基本块在芯片底层执行的流程图，实际在芯片底层执行的流程图如图 C-2 所示。其中 0 号基本块和 3 号基本块均在芯片流水线第 0 级执行，1 号基本块和 2 号基本块均在芯片流水线第 1 级执行。由于资源排布时考虑到了数据依赖和控制依赖，因此虽然最终执行的基本块流程图发生了变

化，但程序原本的逻辑并没有被更改。更改流程图后占用的流水线级数更少，有利于充分利用芯片流水线资源。

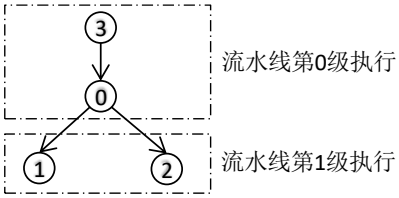

图 C-2　芯片底层实际执行的流程图