Operations Research and Fuzziology 运筹与模糊学, 2024, 14(1), 73-82
Published Online February 2024 in Hans. https://www.hanspub.org/journal/orf
https://doi.org/10.12677/orf.2024.141007

Hans汉斯

改进启发式算法求解PISA架构芯片资源排布问题

宋　耀，张松鸿

贵州大学数学与统计学院，贵州　贵阳

收稿日期：2023年11月2日；录用日期：2023年11月22日；发布日期：2024年2月18日

摘　要

在PISA架构设计时，为减少连线复杂度，往往对流水线各级资源以及各级流水线资源有多种多样约束，研究高资源利用率的资源排布算法对编译器设计尤为重要。论文从资源排布的PISA架构资源约束、流图中基本块约束出发，建立单目标整数规划模型，设计并改进启发式算法并设计近似单调队列求解模型。实验结果表明改进的启发式算法有效降低PISA资源排布方案的复杂度的同时提升了资源利用率。针对两个具体的资源排布问题，给出最小流水层数为59级与35级的资源排布方案，相比基于规则的启发式算法分别降低了21层与15层，验证了算法的有效性与稳定性。

关键词

资源排布，单目标规划，基于规则的启发式算法，近似单调队列，分治算法

Improved Heuristic Algorithm to Solve Resource Allocation Problem of PISA Architecture Chip

Yao Song, Songhong Zhang

School of Mathematics and Statistics, Guizhou University, Guiyang Guizhou

Received: Nov. 2nd, 2023; accepted: Nov. 22nd, 2023; published: Feb. 18th, 2024

Abstract

In the design of PISA architecture, in order to reduce the complexity of connections, there are often various constraints on the resources at all levels of the pipeline, as well as on the resources at

文章引用: 宋耀, 张松鸿. 改进启发式算法求解 PISA 架构芯片资源排布问题[J]. 运筹与模糊学, 2024, 14(1): 73-82.
DOI: 10.12677/orf.2024.141007

all levels of the pipeline. Therefore, studying resource allocation algorithms with high resource utilization is particularly important for compiler design. Starting from the resource constraints of the PISA architecture for resource allocation and the basic block constraints in the flow graph, the paper establishes a single objective integer programming model, designs and improves heuristic algorithms, and designs an approximate monotonic queue solving model. The experimental results show that the improved heuristic algorithm effectively reduces the complexity of PISA resource layout schemes while improving resource utilization. For two specific resource allocation problems, a resource allocation scheme with a minimum number of flow layers of 59 and 35 is proposed. Compared with rule-based heuristic algorithms, it reduces 21 and 15 layers respectively, verifying the effectiveness and stability of the algorithm.

Keywords

Resource Arrangement, Single-Objective Programming, Rule-Based Heuristics Algorithm, Approximate Monotonic Queues, Divide and Conquer Algorithm

1. 引言

随着科技时代发展，计算机、机器人、电动汽车等一系列电子产品，都离不开芯片的支撑[1]。在目前全球经济数字化、科技已成为各国国际战略博弈的重要战场，芯片产业并非仅仅是一个经济问题或者市场问题，成为国家之间用来经济制衡的武器。

传统的交换芯片功能固定，随着网络协议的不断更新，传统交换芯片便不能满足新的需求，重新设计芯片到生产使用又会耗费相当大的时间和人力，为了解决此问题，诞生了可编程的交换芯片 PISA (Protocol Independent Switch Architecture) [2]。PISA 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[3] [4]。

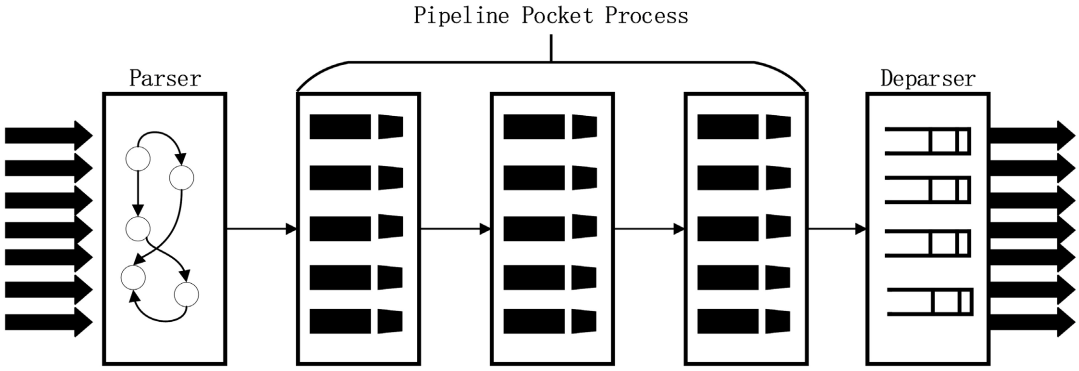

Figure 1. PISA architecture diagram
图 1. PISA 架构图

PISA 架构如图 1 所示，其包括报文解析、多级的报文处理流水线，以及报文重组三个部分。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。在 PISA 架构编

程模型中，使用 P4 语言描述报文处理行为得到 P4 程序。

2. PISA 架构资源排布

PISA 架构芯片资源排布问题中有三个核心的基本概念：报文、基本块、流水线。报文是指网络通信中传输的数据包；基本块是源程序的一个程序片段，对源程序进行基本单元划分会得到一个个的基本块；流水线为一系列处理单元串联构成，报文在流水线中依次通过每个处理单元，最终完成任务处理。

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，基本块在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。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，流水线中针对于这四类资源有着严格的限制资源排布时不能违反芯片的资源限制。

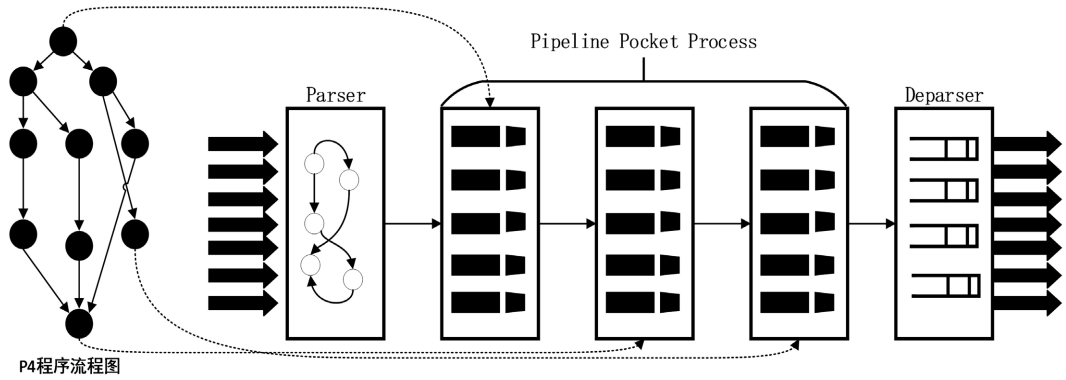

Figure 2. PISA architecture chip resources layout diagram
图 2. PISA 架构芯片资源排布图

3. 资源排布优化模型

假设需要排布基本块数目为 n， A 为一个可行的排布方案，其中 A_{ij} 满足：

$$A_{ij}=\begin{cases}0\\1\end{cases},\left(i,\in[1,n],j\in[0,n-1]\right) \tag{1}$$

其中 $A_{ij}=0$ 表示基本块 i 不排在流水线 j， $A_{ij}=1$ 表示基本块 i 排在流水线 j。由于不能将同一基本块排布到不同的流水线中，因此 A 满足：

$$\sum_{j=0}^{n-1}A_{ij}=1,\left(\forall i,1\le i\le n\right) \tag{2}$$

假设排布方案 A 下，流水线级数为 L，则有：

$$L=f\left(A\right) \tag{3}$$

其中函数 f 定义为排布方案中放置了至少一个基本块的流水线，即 $f\left(A\right)=k$，使得 $\forall i>k$, 都满足：

$$\sum_{j=0}^{n-1} A_{ij} = 0 \tag{4}$$

假设排布过程中，TCAM、HASH、ALU、QUALIFY 四种资源限制与基本块之间的依赖关系所生成的约束空间为V。从而得到优化模型：

$$\min_{A \in V} f(A) \tag{5}$$

求解优化模型等价于找到一个排布方案A^*，使得L最小，即：

$$A^* = \underset{A \in V}{argmin}\, f(A) \tag{6}$$

4. 基于规则的启发式算法

公式(6)所建立的是一个单目标优化模型，Rahman A A 等人使用果蝇优化算法求解驻波热声冰箱电堆单元的单目标优化问题[5]。余璟等人在数学建模中的最优化方法探讨文献中表明求解最优化问题常用算法有梯度下降法、惩罚函数法、遗传算法、蚁群算法等[6]。但由于资源排布时，需要动态考虑控制依赖关系、数据依赖关系、资源限制。上述果蝇优化算法、梯度下降法、惩罚函数法、遗传算法等方法实现复杂度高。董丽薇等人才研究四方物流网络设计问题[7]，尹静等人解决交货期约束的塔式起重机服务调度问题[8]，王军等人在研究施工现场的布局问题[9]，王鹏等人在求解套管柱组合优化问题[10]时采用基于规则的优化算法求解优化问题。虽然启发式算法通常不能求得最优解，但求解大规模问题时，在运算时间上具有优势，且经过合理设计后可以得到满意解[11]。设流水线级数的不可达的下确界值为m，则有$L > m$，m定义为仅考虑资源约束，将所有基本块完全放置下的最小流水线数。据此可以设计如图 3 所示的算法：

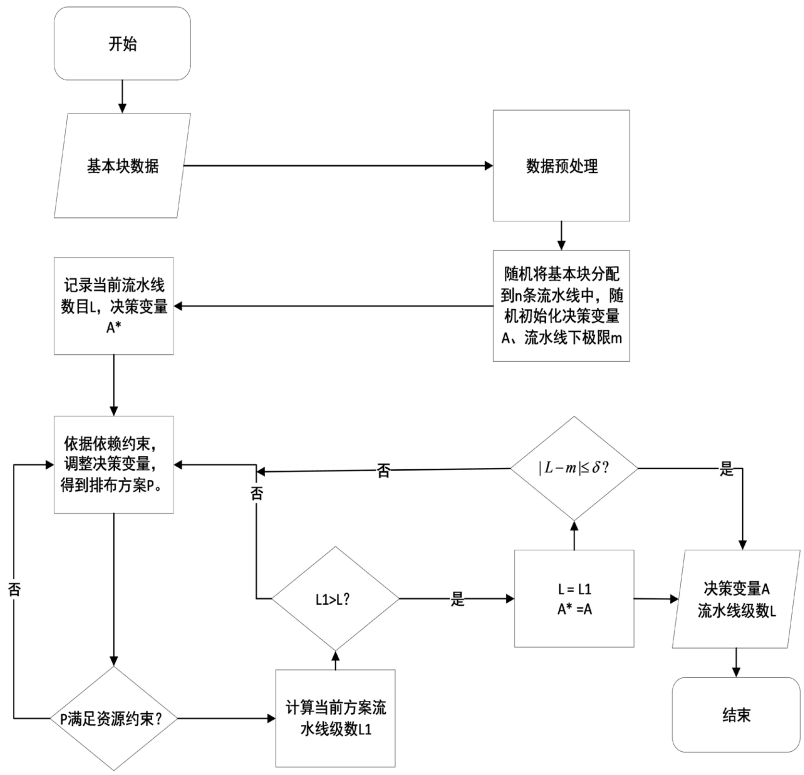

Figure 3. Rule-based heuristics algorithm
图 3. 基于规则的启发式算法

图 3 所示启发式算法中，依据基本块依赖关系约束对排布方案进行调整时，扫描整个排布方案，采取以下两个策略：

1) 若两个基本块之间排布层级不满足依赖关系，则交换彼此排布层次；

2) 若两个基本块之间无任何依赖关系，则在满足资源约束条件下将两个基本块排布到较小的流水层级。

依据数学知识，任何一个变换都可以由有限次对换变换得到，因此总可以在任何一种初始化方案下找到满足资源约束与依赖约束的排布方案。

5. 改进的启发式算法

基于规则的启发式算法在更新决策变量时需要考虑所有的基本块之间的各种依赖关系以及资源约束关系，该过程存在许多重复且没有必要的查询操作。假设 n 个基本块中有 m 个基本块存在依赖关系，则在更新决策变量的过程中，存在至少 $(n-m)^2$ 次没有必要的查询比较操作。因此，在调整基本块位置放置方案时。贺红燕等人在研究集装箱的装载优化问题时，使用分治思想与遗传算法求解装载方案[12]，降低求解装载方案的复杂度。石嵩等人在研究阵列众核处理器调度问题时，使用并改进归并排序算法求解处理器调度排序问题[13]，使得处理器调度更加稳定高效。为此，引入近似单调队列概念，利用分治思想并使用归并排序算法排序[13]，将排序后的结果进行反向扫描调整，最终求解出具有数据依赖或控制依赖的基本块的近似单调队列。

归并排序算法求解得到结果可能会遗漏依赖约束，需要将排序后结果进行反向扫描调整后满足近似单调。举例说明如下：

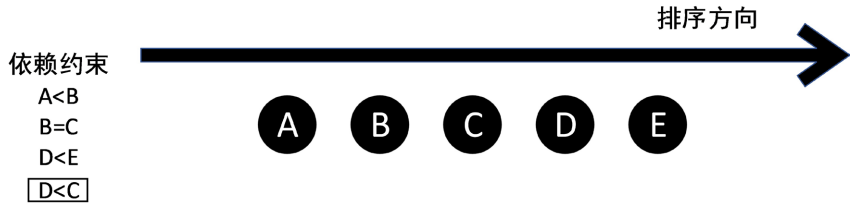

Figure 4. Schematic diagram of sorting omitted dependency constraints
图 4. 排序遗漏依赖约束示意图

由归并排序求解出的结果如图 4 所示。由图 4 中结果可知，当排序方向为从左至右时，由于由基本块构建的流图为有向图，由归并排序进行近似单调排序时，会遗漏图中所示的依赖约束条件 $D<C$。因此增加如图 5 所示的反向扫描操作，找到归并排序中遗漏的约束条件。结合归并排序的结果，加入遗漏依赖约束，调整排序结果。具体如图 6 所示，据遗漏的约束条件需将节点 C 与节点 D 位置对调，最终排序结果才是近似单调队列。

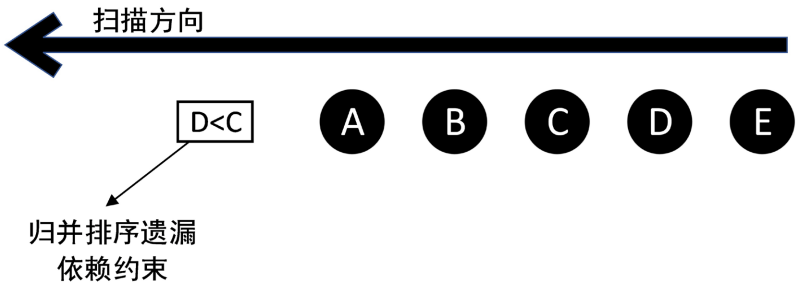

Figure 5. Schematic diagram of reverse scanning to find missing constraints
图 5. 反向扫描查找遗漏约束条件示意图

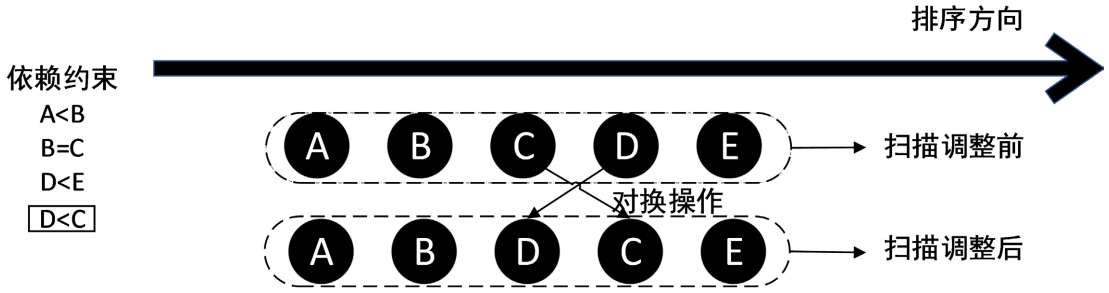

Figure 6. Schematic diagram of the reverse scan search result
图 6. 反向扫描查找结果示意图

据此设计如图 7 所示的改进启发式算法。

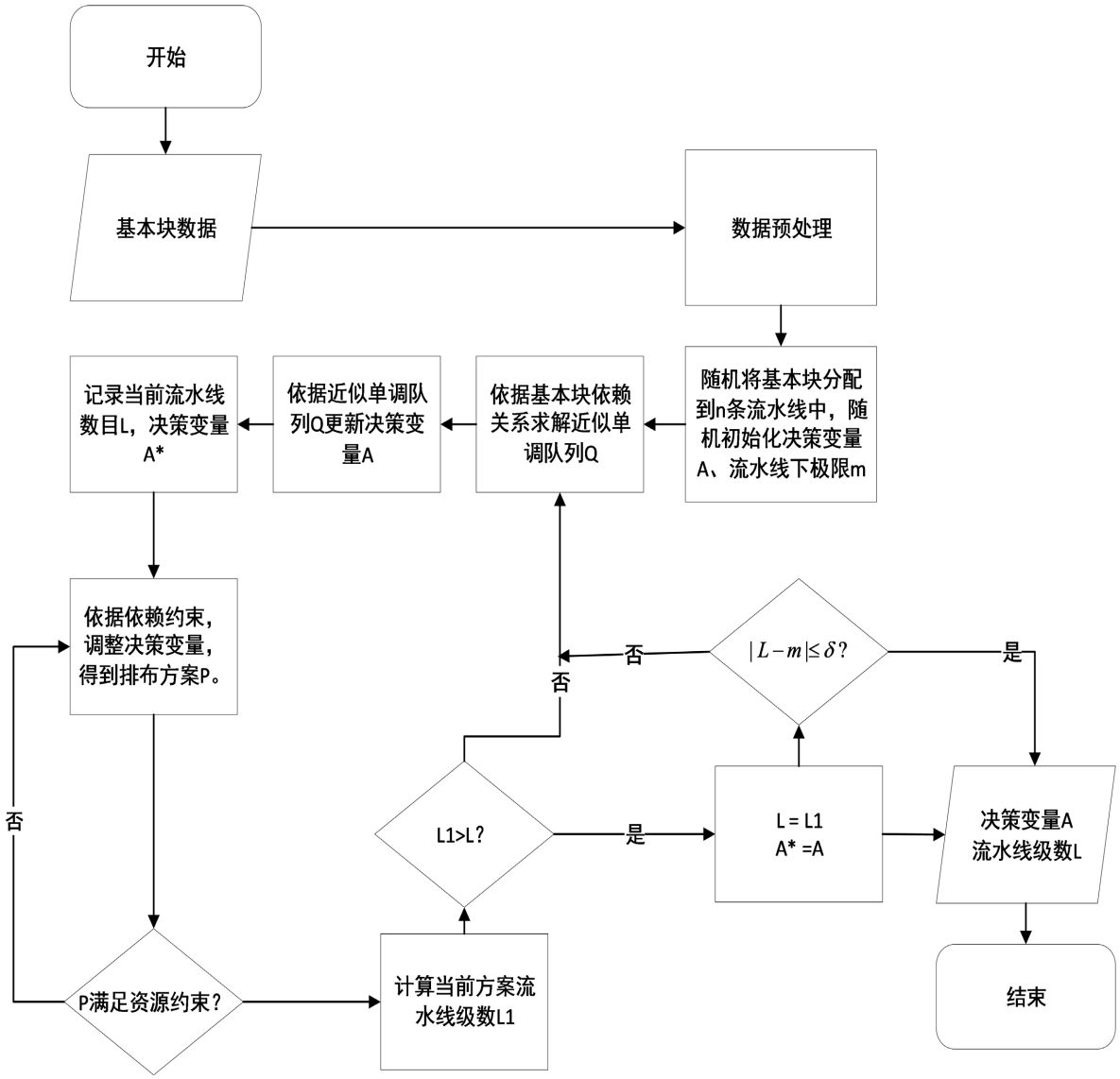

Figure 7. Improved heuristics algorithm
图 7. 改进的启发式算法

6. 数值实验

6.1. 资源约束

本文使用 2022 年中国研究生数学建模 D 题所给数据集进行数值实验。给定资源约束如下：

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。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) 每个基本块只可排布到一级。

依据上述资源约束，CT_i, CH_i, CA_i, CQ_i 分别表示 i 基本块上 TCAM、HASH、ALU、QUALIFY 四种资源上的资源消耗量。则有：

$$\begin{aligned}
&\sum_{i=1}^{n} CT_i A_{ij} \le 1 \\
&\sum_{i=1}^{n} CH_i A_{ij} \le 2 \\
&\sum_{i=1}^{n} CA_i A_{ij} \le 56 \\
&\sum_{i=0}^{n} CQ_i A_{ij} \le 64
\end{aligned} \tag{7}$$

考虑 TCAM 资源在流水线偶数级中限制约束，可得到

$$\sum_{i=1}^{n} \sum_{k=0}^{[n/2]} CT_i A_{i2k} \le 5 \tag{8}$$

6.2. 依赖约束

依赖约束分为控制依赖约束与数据依赖约束。基本块排布到流水线中需要满足基本块间的控制依赖关系，欲在 j 级流水线中放 i 基本块，需要求解该基本块与剩下未排布的基本块的集合 p 控制依赖关系。设集合 p 中基本块放置于流水线上级数集合为 q，则有 $\forall i \in [1,n], j \in [0,n-1], q_k \in q$ 有：

$$\begin{aligned}
&A_{ij}\left(1 - A_{ij} C_{p_k i}\left(j - q_k\right)\right) > 0 \\
&A_{ij}\left(1 - A_{ij} S_{p_k i}\left(j - q_k\right)\right)\left(2 - A_{ij} S_{p_k i}\left(j - q_k\right)\right) > 0
\end{aligned} \tag{9}$$

其中，$C_{p_k i} \in \{0,1\}$，$C_{p_k i} = 0$ 表示基本块 p_k 与基本块之间 i 不存在控制依赖关系 $C_{p_k i} = 1$ 表示基本块 p_k 与基本块 i 之间存在控制依赖关系。$S_{p_k i} \in \{0,1,2\}$，$S_{p_k i} = 0$ 代表基本块 p_k 与基本块 i 之间不存在数据依赖关系。$S_{p_k i} = 1$ 代表基本块 p_k 与基本块 i 之间仅存在读后写数据依赖。$S_{p_k i} = 2$ 代表基本块 p_k 与基本块 i 之间存在写后读或写后写数据依赖。

6.3. 资源约束与依赖约束生成空间

依据上述资源约束与依赖约束可生成约束空间 V_1 如式(10)式所示，式中为方便排布将 $A_{ij} S_{pi}\left(j - q_k\right)$

记为 x。

$$\text{s.t.}\begin{cases}\sum_{j=0}^{n-1} A_{ij}=1,\left(\forall i,1\le i\le n\right)\\ \sum_{i=1}^{n} CT_i A_{ij}\le 1,\left(\forall j,1\le j\le n\right)\\ \sum_{i=1}^{n} CH_i A_{ij}\le 2,\left(\forall j,1\le j\le n\right)\\ \sum_{i=1}^{n} CA_i A_{ij}\le 56,\left(\forall j,1\le j\le n\right)\\ \sum_{i=1}^{n} CQ_i A_{ij}\le 64,\left(\forall j,0\le j\le n\right)\\ \sum_{i=1}^{n} CT_i\left(A_{ij}+A_{ij+16}\right)\le 1,\left(\forall j,0\le j\le 15\right)\\ \sum_{i=1}^{n} CH_i\left(A_{ij}+A_{ij+16}\right)\le 1,\left(\forall j,0\le j\le 15\right)\\ \sum_{i=1}^{n}\sum_{k=0}^{[n/2]} CT_i A_{i2k+1}\le 5\\ A_{ij}\left(1-A_{ij}C_{p_k i}\left(j-q_k\right)\right)>0\\ A_{ij}\left(1-x\right)\left(2-x\right)>0\end{cases} \tag{10}$$

6.4. 实验结果

为进一步验证算法有效性，对 5.1 节资源约束做出修改。首先定义同一执行流程，考虑如图 8 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)。将 5.1 节资源约束第二条修改为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；第三条修改为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。

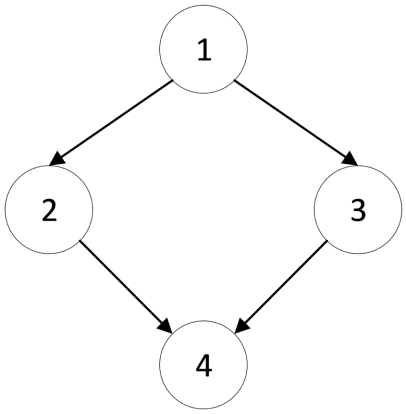

Figure 8. Example of a flow diagram
图 8. 流图示例

依据上述修改，得到资源约束与依赖约束生成空间 V_2 为：

$$\text{s.t.}\begin{cases}\sum_{j=0}^{n-1} A_{ij}=1,\left(\forall i,1\le i\le n\right)\\ \sum_{i=1}^{n} CT_i A_{ij}\le 1,\left(\forall \mathrm{j},0\le j\le n-1\right)\\ HA_j\le 2,\left(\forall \mathrm{j},0\le j\le n-1\right)\\ AL_j\le 56,\left(\forall \mathrm{j},0\le j\le n-1\right)\\ \sum_{i=1}^{n} CQ_i A_{ij}\le 64,\left(\forall \mathrm{j},0\le j\le 15\right)\\ \sum_{i=1}^{n} CT_i\left(A_{ij}+A_{ij+16}\right)\le 1,\left(\forall \mathrm{j},0\le j\le 15\right)\\ HA_j+HA_{j+16}\le 3,\left(\forall \mathrm{j},0\le j\le 15\right)\\ \sum_{i=0}^{n-1}\sum_{k=0}^{[n/2]} CT_i A_{i,2k}\le 5\\ A_{ij}\left(1-A_{ij}C_{pi}\left(j-q_k\right)\right)>0\\ A_{ij}\left(1-x\right)\left(2-x\right)>0\end{cases} \tag{11}$$

式(11)中，HA_j 与 AL_j 分别表示流水线 j 中的最大执行流程中 HASH 资源之和与 ALU 资源之和。基于规则的算法与基于启发式的算法求解得到流水线层级如表 1 所示：

Table 1. Arrangement results of different algorithms
表 1. 不同算法排布结果

约束空间	基于规则的启发式算法	改进的启发式算法
V_1	80	59
V_2	50	35

据实验结果表 1，相较于基于规则的启发式算法，在约束空间 V_1 上，改进的启发式算法排布层数降低了 21 层；在约束空间 V_2 上，改进的启发式算法排布层数降低了 15 层。

7. 总结

论文中模型应用到实际时，需要将考虑实际的资源要求构建约束集，同时需要进一步设计近似单调队列方案以满足超大规模数量级的资源排布问题的时间复杂度要求。此外，该算法仅考虑单目标的优化，在实际应用中需要考虑的不仅仅只有总的资源利用效率这一项，需要考虑不同资源的利用效率，是一个多目标优化问题或者多目标组合优化问题，需要进一步改进算法。从 PISA 架构资源排布问题出发，综合考虑资源排布时各种资源约束，建立基于单目标优化模型并使用改进的启发式算法求解，使得模型具备一定的应用价值。其次，改进的启发式算法对其他类似的有次序依赖的分配排布问题有一定的借鉴作用。

基金项目

贵州省基础研究计划项目(黔科合基础-ZK[2023]一般 038)自然语言语义解析任务的编码方法及高阶优化方法研究，黔科合支撑[2023]一般 432。

参考文献

[1] 刘衡祁. AI 芯片的发展及应用[J]. 电子技术与软件工程, 2019(22): 91-92.

[2] Hang, Z.J., Wen, M., Shi, Y. and Zhang, C.-Y. (2019) Programming Protocol-Independent Packet Processors High-Level Programming (P4HLP): Towards Unified High-Level Programming for a Commodity Programmable Switch. *Electronics*, **8**, Article 958. https://doi.org/10.3390/electronics8090958

[3] Bosshart, P., Daly, D., *et al.* (2014) P4: Programming Protocol-Independent Packet Processors. *ACM SIGCOMM Computer Communication Review*, **44**, 87-95. https://doi.org/10.1145/2656877.2656890

[4] Bosshart, P., Gibb, G., Kim, H.S., *et al.* (2013) Forwarding Metamorphosis: Fast Programmable Match-Action Processing in Hardware for SDN. *ACM SIGCOMM Computer Communication Review*, **43**, 99-110. https://doi.org/10.1145/2534169.2486011

[5] Rahman, A.A. and Zhang, X. (2019) Single-Objective Optimization for Stack Unit of Standing Wave Thermoacoustic Refrigerator through Fruit Fly Optimization Algorithm. *International Journal of Refrigeration*, **98**, 35-41. https://doi.org/10.1016/j.ijrefrig.2018.09.031

[6] 余璟, 岳振军, 贾永兴. 数学建模中的最优化方法探讨[J]. 科技视界, 2021(30): 66-67.

[7] 董丽薇, 黄敏, 匡韩斌, 等. 一种集成最后一公里的四方物流网络设计问题启发式算法[J]. 控制与决策, 2022, 37(6): 1601-1608.

[8] 尹静, 杨阿慧. 考虑交货期约束的塔式起重机服务调度启发式算法[J]. 中国工程机械学报, 2021, 19(1): 1-6.

[9] 王家, 王洋, 邓铁军, 等. 施工现场设施布局优化问题的新型启发式算法[J]. 湖南大学学报(自然科学版), 2020, 47(9): 128-136.

[10] 王鹏, 田懿, 冯定, 等. 基于启发式算法的套管柱组合优化设计方法[J]. 石油钻探技术, 2020, 48(2): 42-48.

[11] 华正明, 杨丽静, 刘振元. 使用启发式算法求解考虑资源可中途加入的资源投入问题[C]//东北大学, 中国自动化学会信息物理系统控制与决策专业委员会, 《控制与决策》编辑部. 第 34 届中国控制与决策会议论文, 2022: 536-541.

[12] 贺红燕, 赵红梅. 基于分治思想和遗传算法的集装箱装载优化模型[J]. 计算机应用与软件, 2021, 38(5): 117-123.

[13] 石嵩, 李宏亮, 朱巍. 阵列众核处理器上的高效归并排序算法[J]. 计算机研究与发展, 2016, 53(2): 362-373.