中国科学: 信息科学 2021 年 第 51 卷 第 5 期: 795–807
SCIENTIA SINICA Informationis
论文

面向光流估计的高效加速器架构设计

刘博生[1,2,3,4], 陈晓明[3,4*], 韩银和[3,4], 常亮[2]

1. 广东工业大学计算机学院, 广州 510006
2. 桂林电子科技大学计算机与信息安全学院广西可信软件重点实验室, 桂林 541004
3. 中国科学院计算技术研究所计算机体系结构国家重点实验室, 北京 100190
4. 中国科学院大学, 北京 100190
* 通信作者. E-mail: chenxiaoming@ict.ac.cn

收稿日期: 2020–05–12; 修回日期: 2020–10–13; 接受日期: 2020–10–15; 网络出版日期: 2021–04–08

国家自然科学基金 (批准号: 61804155, 61834006, U1811264, 61966009, U1711263) 和广西可信软件重点实验室 (批准号: kx202025) 资助项目

摘要 光流 (optical flow) 为同一对象在视频中运动到下一帧的移动量. 从视频中估计光流已广泛应用于各类移动智能系统, 如运动估计和机器人导航. 最近的研究表明, 卷积神经网络 (convolutional neural network, CNN) 能提供可靠的光流估计结果. 然而, 现有的硬件加速器无法支持面向光流估计的 CNN 复杂计算. 具体而言, 这些类型的 CNN 不仅包括常规的卷积 (convolution) 和反卷积 (deconvolution) 运算, 还包括双线性插值 (bilinear interpolation) 和/或关联 (correlation) 运算. 双线性插值和关联操作主要探索两个连续图像帧之间的关联关系. 为解决这一问题, 本项工作提出面向光流的 CNN 硬件加速设计方案 (称为 Swan-AOE), 即通过支持卷积、反卷积、双线性插值和关联操作解决这类神经网络的硬件加速计算问题. Swan-AOE 包括可配置的硬件计算架构和自适应的调度策略, 通过提供灵活的并行调度实现最优化吞吐量计算. 此外, Swan-AOE 还进行设计空间探索, 探索可用片上缓存资源在提高能耗 – 面积效率的潜在能力. 实验结果表明, 与基准加速器相比, 所提出的设计能有效提升性能、能效和面积效率.

关键词 加速器, 光流估计, 能效, 卷积神经网络

1 引言

视频运动分析广泛用于移动智能视觉系统, 如运动估计、移动车辆检测和机器人导航等 [1]. 其中, 由于光流能表示同一对象在视频图像中运动到下一帧的移动量, 光流估计 (optical flow estimation) 成为视频运动分析不可或缺的关键部分. 最近出现的卷积神经网络 (convolutional neural network, CNN) 能在光流估计中提供可靠的精度性能 [2~5]. 尽管如此, 显著的精度提升也带来超大规模计算的巨大代

引用格式: 刘博生, 陈晓明, 韩银和, 等. 面向光流估计的高效加速器架构设计. 中国科学: 信息科学, 2021, 51: 795–807, doi: 10.1360/SSI-2020-0323
Liu B S, Chen X M, Han Y H, et al. Efficient accelerator architecture for optical flow estimation (in Chinese). Sci Sin Inform, 2021, 51: 795–807, doi: 10.1360/SSI-2020-0323

价, 这是由于这些类型的 CNN 需要对两个连续图像帧之间的关联关系进行评估. 具体而言, 面向光流的 CNN 不但需要使用常规卷积 (convolution)/反卷积 (deconvolution) 层进行特征提取 [6, 7], 而且需要通过双线性插值 (bilinear interpolation)/关联 (correlation) 层 [2, 5] 计算连续图像帧之间的对应关系. 以典型的基于光流的 CNN 模型 Flownet [2] 为例, 该模型包括 31 GB 连接, 其计算规模达到图像分类典型模型 Alexnet [8] 的 51 倍. 为克服超大规模计算的难题, 迫切需要提供面向光流 CNN 的硬件加速方案来进行高效计算.

前人工作已针对 CNN 硬件加速器展开深入研究 [9~12]. 然而, 现有的加速器主要集中在加速标准卷积/反卷积计算上, 并且这些加速器不能直接用于面向光流的 CNN 加速计算. 如果要将面向光流的 CNN 模型部署到常规 CNN 加速器上, 则需要借助中央处理器 (central processing unit, CPU) 来处理光流估计, 并同时在加速器上计算标准卷积/反卷积层. 基于我们实验评估得出的结论是, 在 CPU 上进行光流计算的能耗开销比在加速器上进行卷积/反卷积计算的能耗开销高 1 到 2 个数量级. 这意味着光流估计几乎抵消 CNN 加速器带来的高能效效果. 而且, 整个异构系统的能效主要取决于光流估计.

与传统的 CNN 加速器相比, 针对双线性插值和关联操作的硬件加速计算很少受到关注. 现有的双线性插值硬件计算架构 [13] 或者需要大量的硬件实现成本, 或者不支持卷积/反卷积和关联层计算. 尽管关联层的基本操作也是卷积, 现有加速器无法有效处理关联层, 因为关联层中涉及的卷积操作在没有任何权重参数参与的情况下进行. 关联层主要在两个特征图之间进行卷积计算. 作为其中的一种解决方案, 可以将两个加速器分别用于加速常规的卷积/反卷积运算和双线性插值/关联运算. 但是, 由于两个芯片之间需要大量的额外数据传输, 这种直接组合的硬件计算方案效率不高. 综上所述, 目前还没有有效加速光流估计的硬件加速器.

为有效计算光流估计, 本项工作提出一个面向光流 CNN 加速计算的硬件架构, 称为 Swan-AOE. Swan-AOE 支持加速计算面向光流的 CNN 中涉及的 4 种典型网络层, 即卷积层、反卷积层、双线性插值层和关联层. 本项工作分析这 4 类网络层之间的通用操作, 并通过配置和重用硬件资源构建灵活的调度流以解决不同网络层的计算调度问题. 所提出的调度流通过提供多种类型的并行计算和自适应切换, 针对 4 种典型网络层, 能实现最优吞吐量加速计算. 此外, 本项工作发现片上缓存大小对整体计算的能耗 – 面积效率具有很大影响. 通过探索片上缓存容量的不同设计方案, 以权衡能耗和面积成本, 本项工作能实现最佳能耗 – 面积效率.

基于典型光流 CNN 模型的评估结果表明, 与直接将两种不同加速器组合而成的基准相比, 所提出的加速器能实现平均提高 3.54 倍性能, 提升 27.1 倍能效和提高 6.7 倍面积效率. 本文的主要贡献总结如下:

- 据我们所知, 本项工作为首次构建面向光流估计的深度学习加速器设计.
- 本项工作为面向光流的 CNN 所涉及的 4 类典型层设计统一的加速器硬件架构. 通过探索不同网络类型层之间的通用操作, 本项工作可配置和重用硬件资源以在同一硬件架构中支持不同类型的操作.
- 为实现最优吞吐量计算, 本项工作提出灵活的调度流, 实现充分利用硬件资源进行高效计算.
- 为获得最佳的能耗 – 面积效率, 本项工作广泛评估可能的设计方案, 这些方案提供详细的能耗 – 面积效率权衡分析, 以确定片上缓存容量.

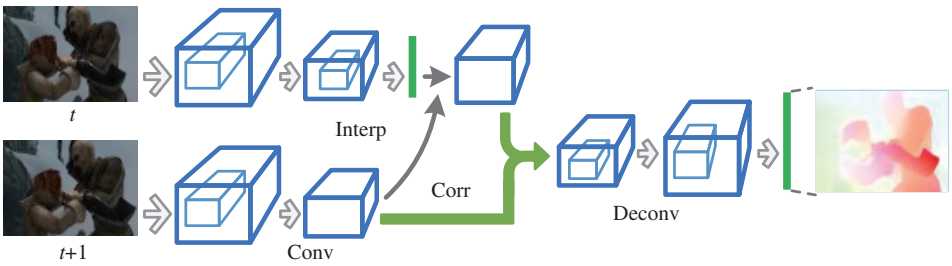

图 1 (网络版彩图) 光流估计系统示意图, 包括卷积 (Conv)、反卷积 (Deconv)、双线性插值 (Interp) 和关联 (Corr) 层

Figure 1 (Color online) Illustration of an optical flow estimation system, which includes convolution (Conv), deconvolution (Deconv), bilinear interpolation (Interp), and correlation (Corr) layers

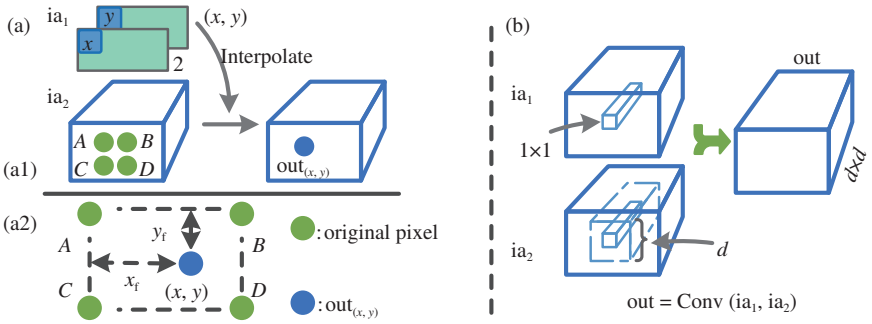

图 2 (网络版彩图) (a) 双线性插值层和 (b) 关联层

Figure 2 (Color online) (a) Bilinear interpolation and (b) correlation layers

2 基础知识

2.1 面向光流估计的 CNN 模型

图 1 介绍用于光流估计的 CNN 基本框架. 该类型的 CNN 以两个连续的图像帧为输入 (即分别在时间 t 和 $t+1$ 的图像), 其最终输出为光流. CNN 主要由卷积 (Conv)、反卷积 (Deconv)、双线性插值 (Interp) 和关联 (Corr) 层组成. 双线性插值层和关联层主要提取两个连续图像帧之间的关联关系. 双线性插值运算由于能在计算效率和图像质量之间提供较好的折衷, 目前已广泛应用于查找两个连续图像帧之间的相关特征 [14]. 另外, 关联层为另一种广泛应用的关联两个连续图像帧的方法 [3].

(a) 双线性插值层. 图 2(a) 为双线性插值计算过程. 其主要根据第一图像帧生成的中间光流 (ia_1) 插值第二图像帧 (ia_2) 的特征, 如图 2(a1) 所示. ia_1 有两个通道. 从 ia_1 的每个通道获取一个像素以形成用于插值的坐标 (x, y). ia_2 的 4 个最近邻像素 (即像素 A, B, C, 和 D) 由 (x, y) 确定. 图 2(a2) 介绍双线性插值计算过程. 双线性插值运算表示为

$$\begin{cases} \mathrm{out}_{(x,y)} = (1-y_\mathrm{f}) \times ((1-x_\mathrm{f}) \times A + x_\mathrm{f} \times B) + y_\mathrm{f} \times ((1-x_\mathrm{f}) \times C + x_\mathrm{f} \times D), \\ x_\mathrm{f} = \mathrm{fraction}(x);\ y_\mathrm{f} = \mathrm{fraction}(y), \end{cases} \tag{1}$$

其中, $\mathrm{out}_{(x,y)}$ 为插值结果, x_f 和 y_f 分别为 x 和 y 的小数部分. 例如, $\mathrm{fraction}(1.3) = 0.3$.

(b) 关联层. 图 2(b) 介绍关联层操作. 该关联操作在没有权重参数的情况下对两个连续图像帧 (ia_1 和 ia_2) 执行卷积运算. 输出映射 (out) 具有 $d \times d$ 个通道, 这些通道来自不同帧中大小为 d 的两个

输入特征图的卷积运算. 尽管关联层也是卷积, 但是不同加速器的直接组合需要大量硬件开销, 导致计算效率不高.

为面向光流的 CNN 提供高效的硬件加速, 更好的方法是无缝地融合不同加速器并构建统一的硬件计算系统. 接下来, 本项工作将讨论针对 4 种典型网络层加速计算的研究动机.

2.2 研究动机

文献 [13] 提出针对双线性插值的硬件加速方案. 基于该方案, 一种加速面向光流 CNN 模型的简单方法是使用两个加速器, 其中一个用于常规卷积/反卷积运算, 另一个用于双线性插值运算.

首先, 通过观察可以发现, 光流估计中涉及的基本操作也可通过加速器中的基本处理单元 (process element, PE) 进行计算. 具体地, 通过在常规的 CNN 加速器中重新使用乘加 (multiplication and accumulation, MAC) 单元完成双线性插值操作. 为充分重用硬件资源以有效支持 4 种类型的典型层, 本项工作提出灵活的调度流, 并针对不同的操作进行优化.

其次, 与标准卷积不同, 关联层计算需要加载来自两个连续图像帧的输入. 由于前人加速器不支持直接加载两组输入神经元, 现有加速器不能平滑地部署关联运算, 这是由于前人加速器主要从神经元和权重参数中获取输入来计算 MAC 操作. 为加速关联层计算, 本项工作在片上缓存和 PE 阵列之间提供灵活互连, 以使加速器能提供来自不同帧的神经元输入数据, 实现支持关联层计算.

最后, 由于卷积和反卷积运算在光流估计中同样重要, 本项工作也重点考虑卷积和反卷积. 由于反卷积层可以通过多个步骤, 平滑地转移到一般卷积运算, 以消除零计算而无需修改 PE 阵列 [11], 本项工作直接在加速器中对反卷积层使用该变换方法. 尽管在前人加速器中已对卷积和反卷积加速进行广泛研究 [9, 10, 12], 但是很少有工作关注片上缓存的效率, 而片上缓存的能耗 – 面积效率在加速器开销中占重要地位.

为实现面向光流 CNN 的高效加速, 本项工作提出新的硬件设计方案. 本项工作首先为 4 个典型层提供不同并行类型的灵活调度流, 以实现高吞吐量计算; 其次, 讨论片上缓存的能耗和面积影响, 通过设计空间探索确定最佳的片上缓存容量. 更具体地, 通过结合所提出的灵活调度流, 本项工作建立最优化问题, 以确定最佳的片上缓存容量大小, 使加速器的能耗 – 面积效率最大化.

3 Swan-AOE 硬件架构和调度流

本节主要介绍所提出加速器的硬件架构和调度流, 包括所提出加速器的基本框架和关键组件, 并介绍针对典型网络层的调度流.

3.1 Swan-AOE 硬件架构

图 3 介绍所提出加速器的整体架构. 该加速器主要包括两个关键组件: 神经处理单元 (neural process element, NPE) 和 3 个片上缓存 (Bin, Bout 和 WB). 其中, 在片上缓存和 NPE 之间通过多路复用器 (MUXes) 实现互连. 通过多路选择, 片上缓存可以提供用于卷积/反卷积所需的权重和神经元, 或用于双线性插值/关联层计算所需的输入神经元组. NPE 包括 3 个关键组件: PE 阵列、双线性插值 (bilinear interpolation, BI) 组件和控制器 (controller, Ctl). PE 阵列包括执行 MAC 操作的 $m \times n$ 个 PE. BI 与 PE 阵列配合执行双线性插值运算. 因此, PE 可重复用于插值运算. Ctl 触发 NPE 正确执行 MAC 操作, 包括将输入加载到 NPE, 在 PE 阵列中进行计算, 并将输出存回片上缓存. 此外, 本项工作还集成直接内存访问 (direct memory access, DMA) 组件, 实现在片外动态随机存取存储器 (dynamic

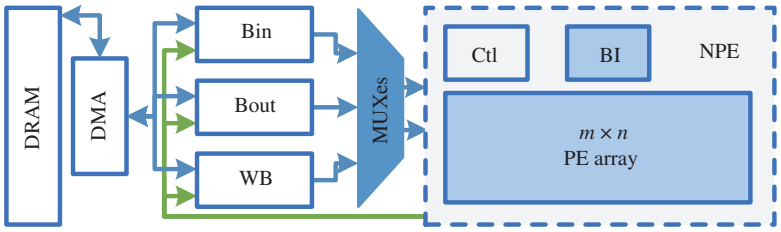

图 3 (网络版彩图) 所提出加速器的硬件架构

Figure 3 (Color online) Overview of the proposed accelerator architecture

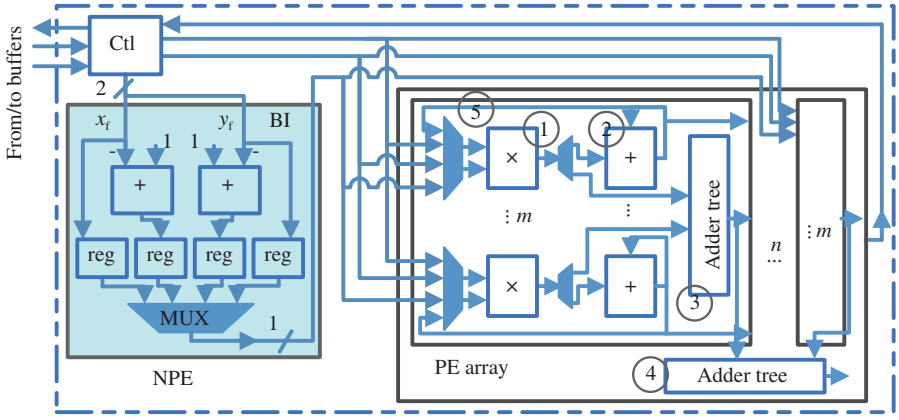

图 4 (网络版彩图) 由 $m \times n$ 个 **PE** 组成的 **NPE** 硬件架构

Figure 4 (Color online) Architecture of NPE composed of $m \times n$ PEs

random access memory, DRAM) 和片上缓存之间传输数据.

图 4 介绍 NPE 硬件架构, 包括 PE 阵列、BI 和 Ctl 组件. PE 阵列包括 $m \times n$ 个乘法器 (①), $m \times n$ 个加法器 (②), n 个 m 输入加法树 (③) 和一个 n 输入加法树 (④). 乘法器从 BI, Ctl 和加法器 (②) 获取输入. 每个 PE 列中乘法器的输出被送到级联的加法器 (②) 或加法树 (③) 进行累加. NPE 支持将 n 个 m 输入加法树 (③) 的输出传输到 n 输入加法树 (④) 以进行进一步累加, 或发送至 Ctl 进行存储. 与用于 MAC 运算的 PE 阵列不同, BI 组件主要执行部分双线性插值运算. 具体而言, BI 从 Ctl 中获取 x_f 和 y_f (x 和 y 的小数部分) 输入, 然后生成式 (1) 中的 $(1-x_f)$, x_f, $(1-y_f)$ 和 y_f. 在此基础上, NPE 将生成结果发送到 PE 阵列进行双线性插值计算.

3.2 调度流

图 5 介绍所提出的灵活调度流. 卷积层的输出特征图 (out, 具有 M 个输出通道) 根据输入特征图 (ia, 具有 N 个输入通道) 和权重参数 (w) 生成. 根据特征图的特征, 卷积层被划分为 3 种类型: 输入特征图很少的层 ($N < n$), 输出特征图很少的层 ($M < m$) 和中间情况 ($N \geqslant n$ 和 $M \geqslant m$). 这三类卷积层分别被图 5(a)~(c) 所处理. 图 5(a) 介绍针对具有较少输入特征图的网络层 ($N < n$) 的调度流. 该调度流在每个执行周期中将一个神经元和 $m \times n$ 个权重作为输入, 然后生成 $m \times n$ 个输出神经元, 如图 5(a1) 所示. 为支持图 5(a1) 的调度流, 图 5(a2) 提供相应的硬件架构. 其通过将图 4 的乘法器 (①) 和加法器 (②) 配置为活动状态实现. 在每个加载周期中, 每个输入神经元由 $m \times n$ 个 PE 共享, 并且 $m \times n$ 个权重被发送到 $m \times n$ 个 PE 进行计算. 因为每个加载周期仅需要一个输入神经元, 所以

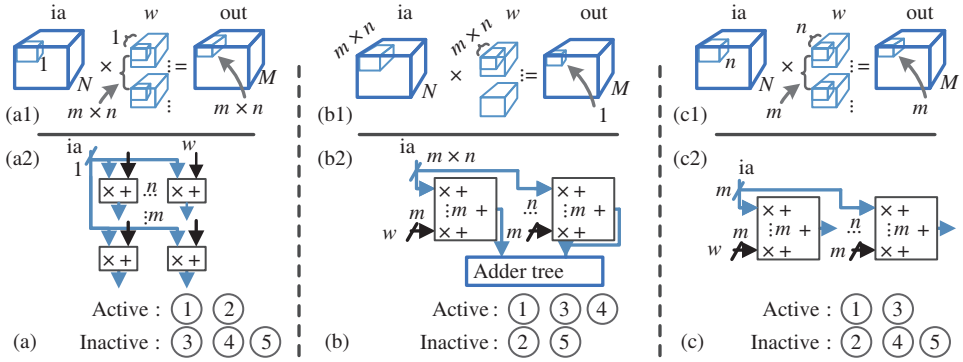

图 5 (网络版彩图) 针对卷积层的调度流: (a) 少量输入特征图 ($N < n$), (b) 少量输出特征图 ($M < m$) 和 (c) 中间情况 ($N \geqslant n$ 和 $M \geqslant m$). M 和 N 分别为输入和输出通道. m 和 n 分别为 PE 阵列的行和列号. (a1)~(c1) 为映射方案, 而 (a2)~(c2) 为 NPE 对应的活跃模块

Figure 5 (Color online) Schedules for convolution operations with (a) few input maps ($N < n$), (b) few output maps ($M < m$), and (c) the middle case ($N \geqslant n$ and $M \geqslant m$). M and N are respectively the input and output channels. m and n are the row and column numbers of the PE array, respectively. (a1)~(c1) are the mapping schemes, while (a2)~(c2) are the active architectures of the NPE component

当卷积层具有很少的输入特征图时, 所提出的加速器仍能充分利用 PE 阵列进行加速计算.

与图 5(a) 的处理具有较少输入特征图的调度不同, 图 5(b) 介绍针对具有少输出特征图 ($M < m$) 的调度流. 该调度以 $m \times n$ 个神经元和权重作为输入, 然后在每个执行周期生成一个输出神经元, 如图 5(b1) 所示. 为支持所提出的调度流, 需要将乘法器 (①), 加法树 (③和 ④) 配置为活动状态, 如图 5(b2) 所示. $m \times n$ 个神经元和权重被加载到 PE 阵列, PE 阵列通过加法树生成一个输出. 由于每个周期最多产生一个输出神经元, 所提出的加速器能充分利用 PE 阵列处理具有较少输出特征图的卷积层.

不同于针对具有较少输入或输出特征图的调度流, 图 5(c) 介绍针对中间情况 ($N \geqslant n$ 和 $M \geqslant m$) 的调度流解决方案. 图 5(c1) 中的调度流将 n 个输入神经元和 $m \times n$ 个权重 (m 个输出通道和 n 个输入通道) 作为输入, 然后生成 m 个具有不同通道的输出神经元. 图 5(c2) 介绍相应的 PE 架构. 该架构由图 4 中的①和 ③配置为活动状态实现. 图 5(c) 的调度流与文献 [9] 相似, 后者通过加载多通道权重和神经元执行并行操作. 和前人工作不同之处在于, 本项工作提供灵活的调度流和可配置的 PE 阵列架构来最优化计算吞吐量.

(a) 双线性插值映射方案. 图 6 举例介绍用于双线性插值的映射方案. PE 阵列中用于支持该映射方案的活动硬件模块类似于图 5(a). 与处理权重和神经元的图 5(a) 不同, 图 6(a) 的调度流着重计算输入神经元组, 主要通过 BI 和 PE 阵列协作完成计算. 具体而言, 该调度流将来自不同通道的 (x, y) 和 $m \times n$ 个相应的神经元作为输入, 如图 6(a) 所示, 然后在每个执行周期最多生成 $m \times n$ 个输出神经元. 图 6(b) 介绍双线性插值的详细计算过程. 例如, 在周期 1 时, NPE 将来自 BI 的 $(1 - x_{\mathrm{f}})$ 和不同输入通道的 $m \times n$ 个 A 添加到 PE 阵列进行计算. 乘法器 (①) 和加法器 (②) 被配置为活跃状态, 用于执行 MAC 操作. 类似地, 将式 (1) 的 4 个邻近点按顺序添加到 PE 阵列以进行 MAC 操作. 由于来自不同行的邻近点被按顺序添加, 该调度流可避免直接从输入特征图的不同行加载输入.

(b) 关联层映射方案. 图 7 举例介绍针对关联层的映射方案. PE 阵列中用于支持该映射方案的硬件活动模块类似于图 5(b). 但是, 与以权重和神经元组作为输入的图 5(b) 不同, 关联层的调度流针对输入神经元组执行 MAC 操作. 该调度流将两组神经元 (每组 $m \times n$ 个) 作为输入, 然后在每个执行

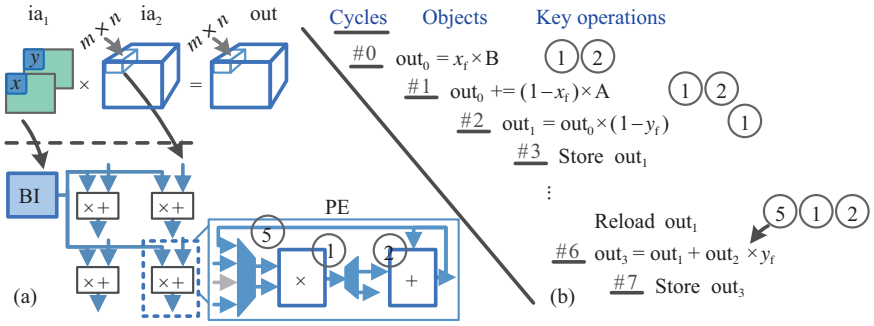

图 6 (网络版彩图) 双线性插值映射方案. (a) 映射; (b) 计算过程

Figure 6 (Color online) Mapping scheme of bilinear interpolation. (a) Mapping; (b) computing process

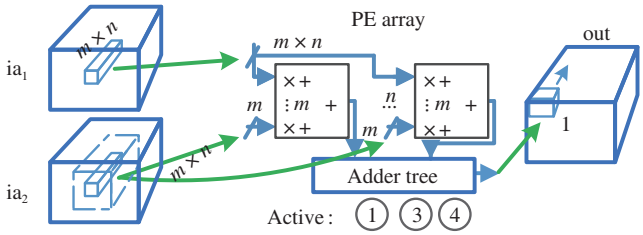

图 7 (网络版彩图) 关联层映射方案

Figure 7 (Color online) Mapping scheme of the correlation operation

周期最多生成一个输出神经元. 为支持该调度流, 需要将图 4 中的 ①, ③和 ④硬件模块配置为活动状态. 具体地, 每个 PE 列的 m 个乘法器 (①) 输出被送到 m 个输入加法树 (④). 在此之后, 将 n 个 m 输入加法树的输出通过 n 输入加法树 (④) 相加在一起得到最终输出.

由于针对不同典型层的调度流需要的数据输入布局不同, 所设计的加速器需要逐层准备输入和输出数据. 为支持这一功能, Ctl 需要在 MAC 操作之前和之后创建正确的数据布局. 具体而言, Ctl 只需拆分输入神经元或合并输出神经元, 因为输入神经元始终来自不同的输入通道, 因此可以在不同的调度之间动态切换. 在每个执行周期, Ctl 支持将 1, n 或 $m \times n$ 个输入神经元发送到 PE 阵列以进行 MAC 操作. 同时, Ctl 支持收集 1, m 或 $m \times n$ 个输出神经元进行存储. 另外, 由于所提出加速器仅用于推理过程, 权重参数重组可以在推理之前完成. 因此, 加速器始终可以正确执行 MAC 操作以进行光流估计.

4 设计空间探索

本节介绍针对片上缓存容量大小的设计空间探索, 以实现最优化能耗 – 面积效率. 本节首先针对片上缓存容量建立最优化问题, 然后结合调度流和最优化问题确定片上缓存的最佳容量大小.

4.1 最优化片上缓存容量

本项工作需要优化的参数分别为片上缓存 Bin, Bout 和 WB 的容量大小: z_{IB}, z_{OB} 和 z_{WB}. 通过在可用的片上缓存资源下探索 z_{IB}, z_{OB} 和 z_{WB} 的可能组合, 加速器能实现最优能耗 – 面积效率. 由于 z_{IB}, z_{OB} 和 z_{WB} 的不同组合导致不同的能耗和面积开销, 这里使用能耗和面积的欧几里德范数来

评估综合效率. 对于 z_{IB}, z_{OB} 和 z_{WB} 的所有可能组合, 加速器的最佳能耗 – 面积效率表示为

$$\min \ \sqrt{f^2_{\mathrm{energy}}(z_{\mathrm{IB}}, z_{\mathrm{OB}}, z_{\mathrm{WB}}) + f^2_{\mathrm{area}}(z_{\mathrm{IB}}, z_{\mathrm{OB}}, z_{\mathrm{WB}})}, \tag{2}$$

其具有以下约束:

$$z_{\mathrm{IB}} = z_{\mathrm{OB}}, \tag{3}$$

$$z_{\mathrm{IB}}, z_{\mathrm{OB}}, z_{\mathrm{WB}} \in \{8, 16, 32, 64, 128, 256, 512\} \ \mathrm{kB}, \tag{4}$$

f_{energy} 和 f_{area} 分别为加速器的总能耗和总面积, 在式 (5) 和 (6) 中定义. 约束 (3) 和 (4) 有以下特性:

- 约束 (3) 设置 z_{IB} 等于 z_{OB}. 这是因为 Bin 和 Bout 缓存交替存储输入神经元和输出神经元.
- 约束 (4) 确定 z_{IB}, z_{OB} 和 z_{WB} 可以选择的容量范围.

式 (2) 中加速器的能耗和面积开销表示为

$$f_{\mathrm{energy}}(z_{\mathrm{IB}}, z_{\mathrm{OB}}, z_{\mathrm{WB}}) = f_{\mathrm{sram\_energy}}(z_{\mathrm{IB}}, z_{\mathrm{OB}}, z_{\mathrm{WB}}) + f_{\mathrm{npe\_energy}} + f_{\mathrm{dram\_energy}}(z_{\mathrm{IB}}, z_{\mathrm{OB}}, z_{\mathrm{WB}}), \tag{5}$$

$$f_{\mathrm{area}}(z_{\mathrm{IB}}, z_{\mathrm{OB}}, z_{\mathrm{WB}}) = f_{\mathrm{sram\_area}}(z_{\mathrm{IB}}, z_{\mathrm{OB}}, z_{\mathrm{WB}}) + f_{\mathrm{npe\_area}}, \tag{6}$$

其中 $f_{\mathrm{sram\_energy}}$, $f_{\mathrm{npe\_energy}}$ 和 $f_{\mathrm{dram\_energy}}$ 分别为片上缓存, NPE 和片外 DRAM 的能耗, 这些能耗根据所提出的加速器进行评估. 式 (6) 的 $f_{\mathrm{sram\_area}}$ 和 $f_{\mathrm{npe\_area}}$ 分别为片上缓存和 NPE 的面积. 式 (6) 中不包括 DRAM 的面积, 因为 DRAM 不属于加速器的一部分.

通过建立最优化问题 (式 (2)∼(6)), 本项工作能找到最佳的片上缓存参数 z_{IB}, z_{OB} 和 z_{WB}, 从而产生最优的能效 – 面积效率.

4.2 参数优化求解过程

由于 z_{IB}, z_{OB} 和 z_{WB} 的选择有限, 可通过穷举搜索式 (2)∼(6) 解决组合优化问题. 通过观察发现, 仅基于面向单个光流的 CNN 模型进行搜索不能很好适应其他网络模型. 为获得最佳的能耗 – 面积效率, 本项工作基于多个典型的光流 CNN 模型 (本项工作采用 3 个代表性的 CNN) 进行设计空间探索. 并且, 所使用网络模型结果的几何平均值用于搜索空间评估. 通过基于不同光流 CNN 模型进行逐层搜索, 以找出最佳的 z_{IB}, z_{OB} 和 z_{WB} 参数值, 最终实现优化能耗 – 面积效率.

5 实验评估

5.1 实验设置

本项工作在 Verilog 中实现所提出的加速器. 首先, 通过使用基于 65 nm 工艺的 Design Compiler 对所实现的加速器进行综合, 然后使用 PrimeTime-PX 评估所设计加速器的功耗. 片上缓存由 Memory Compiler 生成. 片外缓存由 CACTI [15] 工具进行评估. 同时, 本项工作构建一个周期级的模拟器来评估性能. 使用的面向光流的 CNN 模型包括 Flownet [2], Spynet [4] 和 Pwcnet [5]. 其中, Flownet 为第一个提出用于光流估计的深度卷积模型. Spynet 将深度卷积运算与经典的金字塔网络模型相结合, 以解决光流估计中存在的模型尺寸问题. Pwcnet 结合不同的关联方法, 即双线性插值层和关联层, 以处理两个连续的图像帧. 表 1 列出面向光流 CNN 模型的详细特性, 使用的数据集为 KITTI [16].

本项工作分两个步骤对加速器进行评估. 首先, 本项工作基于 CNN 典型模型对片上缓存大小进行设计空间探索. 其次, 本项工作将所提出的加速器与最先进的前人加速器基准进行比较. 由于

表 1 基于光流 CNN 的特性

Table 1 Characteristics of optical flow oriented CNNs

CNNs	Bilinear interpolation	Correlation	Total layers
Flownet [2]	0	1	27
Spynet [4]	4	0	33
Pwcnet [5]	4	5	89

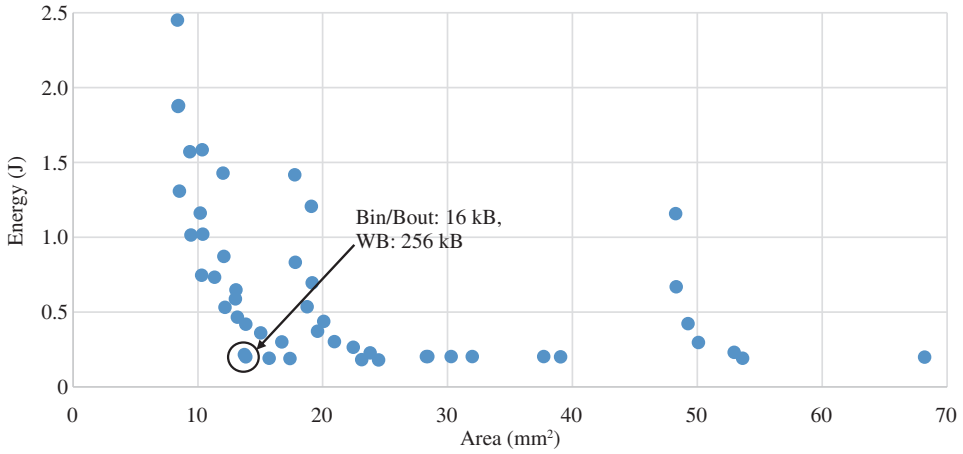

图 8 (网络版彩图) 不同片上缓存大小下, 所提出设计的能耗和面积开销趋势

Figure 8 (Color online) Energy and area trend of the proposed design under different on-chip buffer sizes

前人工作中没有用于光流估计的专用加速器, 本项工作构建一个包括两个部分的组合加速器基准 (用 "Eyeriss-plus" 表示): Eyeriss 加速器 [10] 和双线性插值加速器 [13]. Eyeriss 用于计算传统卷积和反卷积. 另外, 关联层计算主要借助片外缓存将神经元移至权重缓存, 实现在组合加速器基准中支持关联层计算. 由于文献 [13] 为基于现场可编程逻辑门阵列 (field programmable gate array, FPGA) 的解决方案, 本项工作实现其硬件架构, 并将片上缓存和乘法器调整为与加速器具有相同规模以进行公平比较.

5.2 仿真结果

(a) 设计空间探索. 本项工作基于 16×16 的 PE 阵列进行片上缓存 (Bin/Bout/WB) 容量大小的设计空间探索. 每个片上缓存的容量变化范围为 8 kB 到 512 kB.

图 8 介绍在不同片上缓存大小下, 所提出加速器的能耗和面积开销趋势. 可以看出, 当 Bin = Bout = 16 kB 和 WB = 256 kB 时, 所提出的设计能实现能耗和面积效率的最佳平衡. 同样, 当所提出的设计具有非常小的片上缓存时 (图中处于较小面积的区域), 由于大量额外的片外缓存, 能耗开销大大增加.

图 9 介绍在不同缓存大小下加速器的能效趋势. 可以发现, 当 Bin = Bout = 16 kB 和 WB = 256 kB 时, 加速器也达到近似最优的能效. 另外, 由于片上缓存自身的延迟影响, 当总缓存大小大于 800 kB 时, 加速器的能效会降低. 基于以上针对片上缓存容量的设计空间探索, 本项工作使用最佳的片上缓存容量大小 (片上缓存总大小为 288 kB, Bin = Bout = 16 kB 和 WB = 256 kB).

(b) 性能. 为进行公平的性能比较, 本项工作通过 Scale-Sim 模拟器 [17] 将 Eyeriss 的 PE 规模和片上缓存大小扩展到与所提出加速器相同的规模. 图 10 介绍归一化的性能比较. Swan-AOE 的平均

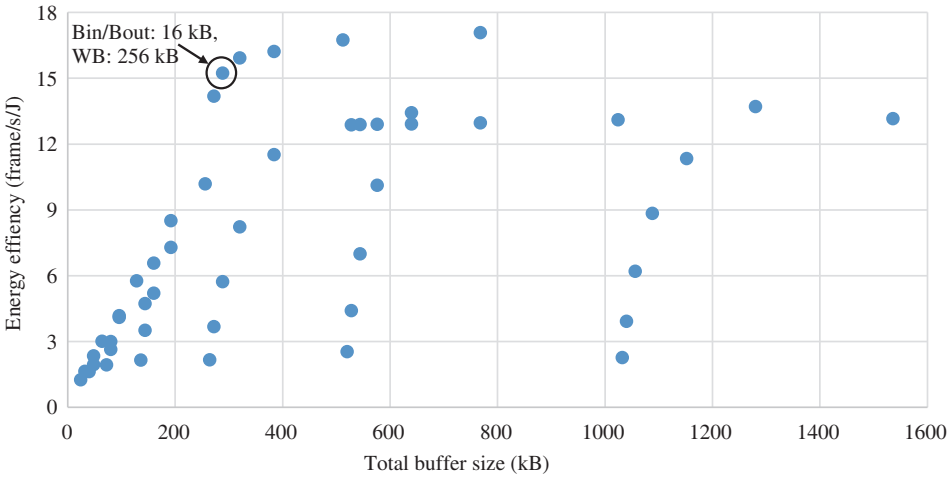

图 9 (网络版彩图) 不同片上缓存大小下的能效趋势

Figure 9 (Color online) Energy efficiency trend under different on-chip buffer sizes

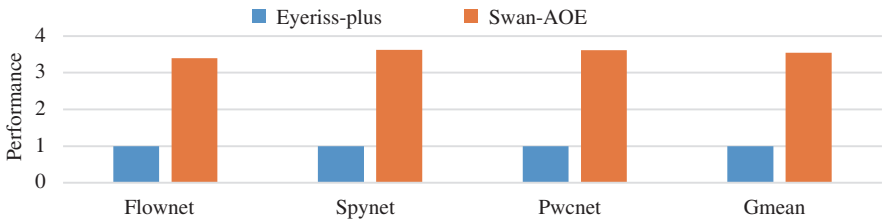

图 10 (网络版彩图) 与基准进行归一化的性能比较. "Gmean" 表示性能的几何平均值

Figure 10 (Color online) Normalized performance comparison with the baseline. "Gmean" denotes the geometrical mean of the performance

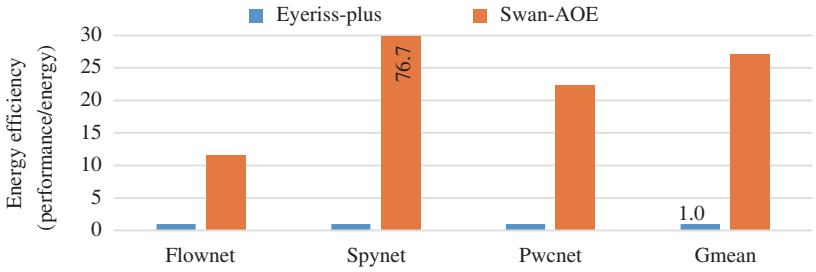

图 11 (网络版彩图) 正则化能效比较

Figure 11 (Color online) Normalized energy efficiency comparison

性能比基准高 3.54 倍.

(c) 能效. 图 11 介绍归一化的能效比较, 该比较基于加速器的原始 PE 尺寸 (Swan-AOE 的 PE 规模是 Eyeriss 的 1.52 倍). 与基准相比, Swan-AOE 的能效提高 27.1 倍. Swan-AOE 获得更高能效, 主要归于高效的片上缓存能有效减少片外缓存访问. 例如, Eyeriss 在卷积和反卷积操作中的能耗比所提出加速器的能耗高 6.6 倍. Swan-AOE 在 Spynet 获得最高能效 (76.7 倍) 源于性能提升和设计空间探索减少的能耗开销.

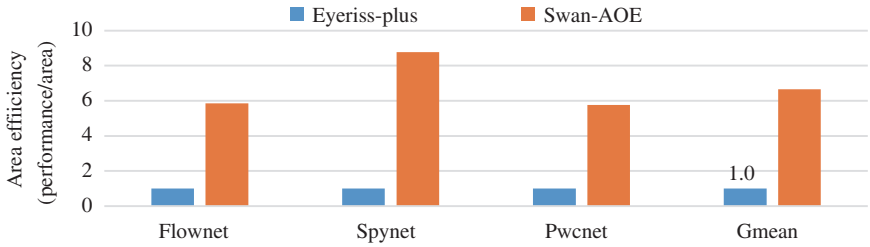

图 12 (网络版彩图) 正则化面积效率比较

Figure 12 (Color online) Normalized area efficiency comparison

表 2 提出的加速器明细

Table 2 Breakdown of the proposed accelerator

	Technology (nm)	# of PEs	Frequency (MHz)	Power (mW)	Area (mm^2)	Peak performance (GOP/s)	Peak power efficiency (GOP/s/W)
Eyeriss	65	168	200	278	12.25	67.2	241.7
Swan-AOE	65	256	700	405.8	13.8	548.8	1352.4

(d) 面积效率. 图 12 介绍归一化的面积效率比较. 该比较在加速器的原始 PE 尺寸下进行. Swan-AOE 的面积效率比基准高 6.7 倍. 所提高的面积效率来自设计空间探索和光流估计的性能提升. 另外, Eyeriss 占基准总面积的 60.1%.

(e) 综合比较. 表 2 列出所提出加速器的功耗和面积明细. 与 Eyeriss 相比, 由于本项工作所设计的 PE 规模较大 (1.5 倍), 所提出的设计需要消耗更多的功耗和面积 (分别为 1.5 倍和 1.1 倍). 具体而言, 所提出加速器的片上缓存占总功耗的 30.7% 和总面积的 54.2%. 但是, 由于具有有效的光流计算能力, 与仅面向 CNN 的加速器相比, 所提出的设计可实现更高的能耗 – 面积效率.

6 结论

本项工作提出一个面向光流估计的加速器硬件架构, 该架构可实现显著的能耗 – 面积效率. 本项工作引入灵活的调度流, 以充分利用硬件资源来实现高效的光流加速计算. 同时, 本项工作通过设计空间探索确定片上缓存大小, 实现获得最佳能耗 – 面积效率. 评估结果表明, 与两个不同加速器的组合基准相比, 所提出的设计实现 3.54 倍的更高性能, 27.1 倍的能效和 6.7 倍的面积效率.

参考文献

1 Menze M, Geiger A. Object scene flow for autonomous vehicles. In: Proceedings of IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2015. 3061–3070
2 Dosovitskiy A, Fischer P, Ilg E, et al. Flownet: learning optical flow with convolutional networks. In: Proceedings of IEEE International Conference on Computer Vision (ICCV), 2015. 2758–2766
3 Ilg E, Mayer N, Saikia T, et al. Flownet 2.0: evolution of optical flow estimation with deep networks. In: Proceedings of IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2017. 2462–2470
4 Ranjan A, Black M J. Optical flow estimation using a spatial pyramid network. In: Proceedings of IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2017. 4161–4170
5 Sun D, Yang X, Liu M Y, et al. PWC-Net: CNNs for optical flow using pyramid, warping, and cost volume. In: Proceedings of IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2018. 8934–8943

6 Szegedy C, Liu W, Jia Y, et al. Going deeper with convolutions. In: Proceedings of IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2015. 1–9
7 Noh H, Hong S, Han B. Learning deconvolution network for semantic segmentation. In: Proceedings of IEEE International Conference on Computer Vision (ICCV), 2015. 1520–1528
8 Krizhevsky A, Sutskever I, Hinton G E. ImageNet classification with deep convolutional neural networks. In: Proceedings of the 25th International Conference on Neural Information Processing Systems, 2012. 1097–1105
9 Chen T, Du Z, Sun N, et al. DianNao: a small-footprint high-throughput accelerator for ubiquitous machine-learning. SIGARCH Comput Archit News, 2014, 42: 269–284
10 Chen Y H, Krishna T, Emer J S, et al. Eyeriss: an energy-efficient reconfigurable accelerator for deep convolutional neural networks. IEEE J Solid-State Circ, 2017, 52: 127–138
11 Feng Y, Whatmough P, Zhu Y. ASV: accelerated stereo vision system. In: Proceedings of the 52nd Annual IEEE/ACM International Symposium on Microarchitecture, 2019. 643–656
12 Liu B S, Chen X M, Han Y H, et al. Accelerating DNN-based 3D point cloud processing for mobile computing. Sci China Inf Sci, 2019, 62: 212206
13 Banz C, Hesselbarth S, Flatt H, et al. Real-time stereo vision system using semi-global matching disparity estimation: architecture and FPGA-implementation. In: Proceedings of International Conference on Embedded Computer Systems: Architectures, Modeling and Simulation, 2010. 93–101
14 Zhang Y B, Zhao D B, Zhang J, et al. Interpolation-dependent image downsampling. IEEE Trans Image Process, 2011, 20: 3291–3296
15 Muralimanohar N, Balasubramonian R, Jouppi N. Optimizing NUCA organizations and wiring alternatives for large caches with CACTI 6.0. In: Proceedings of IEEE/ACM International Symposium on Microarchitecture, 2007. 3–14
16 Geiger A, Lenz P, Stiller C, et al. Vision meets robotics: the KITTI dataset. Int J Robot Res, 2013, 32: 1231–1237
17 Samajdar A, Zhu Y, Whatmough P, et al. SCALE-sim: systolic CNN accelerator simulator. 2018. ArXiv:1811.02883

Efficient accelerator architecture for optical flow estimation

Bosheng LIU[1,2,3,4], Xiaoming CHEN[3,4*], Yinhe HAN[3,4] & Liang CHANG[2]

1. *School of Computer Science and Technology, Guangdong University of Technology, Guangzhou* 510006*, China*;
2. *Guangxi Key Laboratory of Trusted Software, School of Computer Science and Information Security, Guilin University of Electronic Technology, Guilin* 541004*, China*;
3. *State Key Laboratory of Computer Architecture, Institute of Computing Technology, Chinese Academy of Sciences, Beijing* 100190*, China*;
4. *University of Chinese Academy of Sciences, Beijing* 100190*, China*
* Corresponding author. E-mail: chenxiaoming@ict.ac.cn

Abstract Optical flow plays a key role in recording the movement of an object in videos to its next frame. Estimating optical flow from a video has been widely applied in many mobile cognitive systems such as motion estimation and robot navigation. Recent works have demonstrated that optical flow estimation can be successfully solved by emerging convolutional neural networks (CNNs). Existing dedicated accelerators, however, cannot support the complex computation of optical flow oriented CNNs. Specifically, these CNNs are composed of not only the conventional convolution and deconvolution operations, but also the bilinear interpolation and/or correlation operations, which exploit the correspondence of two consecutive image frames. To address this problem, we propose an integrated accelerating solution called Swan-AOE aiming to tackle optical flow oriented CNNs, i.e., by supporting convolution, deconvolution, bilinear interpolation, and correlation layers. Swan-AOE provides a configurable architecture together with an adaptive schedule that offers multiple types of parallelism to achieve the optimal throughput. In addition, Swan-AOE introduces a design space exploration of the potential of available on-chip memory resources for energy-area efficiency. Experimental results show that the proposed design can efficiently improve the performance, energy efficiency, and area efficiency compared with a comparable combined accelerator baseline.

Keywords accelerator, optical flow estimation, energy efficiency, convolutional neural networks

Bosheng LIU received the Ph.D. degrees in computer science from the Institute of Computing Technology, Chinese Academy of Sciences, Beijing, China, in 2020. He is now an assistant professor with the School of Computer Science and Technology, Guangdong University of Technology, China. His research interests include embedded DNN accelerator and reconfigurable computing.

Yinhe HAN received the M.S. and Ph.D. degrees in computer science from Institute of Computing Technology, Chinese Academy of Sciences, in 2003 and 2006, respectively. He is currently a professor at the Institute of Computing Technology, Chinese Academy of Sciences, His research interests include VLSI/SOC interconnection, testing and fault-tolerance.

Xiaoming CHEN received the B.S. and Ph.D. degrees in electronic engineering from Tsinghua University, Beijing, China, in 2009 and 2014, respectively. He is now an associate professor with the Institute of Computing Technology, Chinese Academy of Sciences. His current research interests include computer-aided design for integrated circuits and advanced computer architecture design.

Liang CHANG received the Ph.D. degree in computer science from the Institute of Computing Technology, Chinese Academy of Sciences, Beijing, China. He is currently a professor with the School of Computer Science and Information Security, Guilin University of Electronic Technology, Guilin, China. His research interests include information security, knowledge representation and reasoning, description logics, and the semantic Web.