

多项式数据通路的优化方法

李东海 朱晓晨 范中磊 杨小军
(长安大学信息工程学院 西安 710064)

摘 要 为了实现多项式数据通路的高层次综合,采用有序的、简化的和正则的带权值广义表模型表达该多项式。提出了基于带权值广义表的优化方法,该方法以自顶向下的方式遍历带权值广义表中的节点,迭代地识别其相应的加法割和乘法割,进而形成一个可允许割序列;根据可允许割序列产生相应可调度的数据流图。实验结果表明,采用该方法得到的数据流图与已有方法得到的相比,在延迟上具有一定的优势。

关键词 多项式,带权值广义表,数据通路,高级综合

中图法分类号 TP391.7 **文献标识码** A

Method of Optimization for Polynomial Datapaths

LI Dong-hai ZHU Xiao-chen FAN Zhong-lei YANG Xiao-jun
(School of Information Engineering, Chang'an University, Xi'an 710064, China)

Abstract In order to implement the high level synthesis of polynomial datapaths, the ordered, reduced and canonical weighted generalized list was used to represent the polynomial. Based on the weighted generalized list, a method for optimizing polynomial datapaths was proposed, which traverses the nodes of the weighted generalized list in a top-down fashion and identifies the corresponding additive cuts and multiplicative cuts iteratively, and then an admissible cut sequence is generated; according to the admissible cut sequence, the corresponding schedulable data flow graph is obtained. Experimental results demonstrate the superiority of the schedulable data flow graph obtained by the proposed method in the latency compared with the existent methods.

Keywords Polynomial, Weighted generalized list, Datapaths, High level synthesis

在高层次设计规范中经常遇到表达为多项式的计算,这些计算主要应用于计算机图形设计和数字信号处理(Digital Signal Processing, DSP),例如数字滤波和 DSP 转换,其中相应的设计规范为采用 C/C++ 编写的算法。为了处理这种抽象的描述,设计者在高级综合之前需要有效的优化工具对初始的规范编码进行优化,而传统的高级综合方法没有对该任务提供足够的支持,并且高层次综合中的结构优化技术(例如调度、资源分配和绑定)不涉及前端的算术优化。这些方法直接从原始设计规范中转换出相应的表达^[1-4],但缺少对规范可能的修改,结果使得结构优化的范围严重受限。

文献[5]提出了在高级综合和硬件描述语言编译器中尝试实现优化转换方法,该方法主要依赖于代数基本性质(例如结合律、交换率和分配律)对代数表达式进行操作,但这些方法没有提供一个系统的方式对初始设计规范进行优化,因而未得到最佳的数据流图(Data Flow Graph, DFG)。一些高级综合系统,例如 Cyber 和 Catapult C,虽然采用了许多代码优化方法(基于核的代数因子分解、分枝定界、推测代码迁移、死码消除等)^[6,7],但是它们并没有依赖于任何能够保证局部最优性转换的正则表达式。文献[8-10]采用基于核的分解算法,该算法采用代数方法来减少多项式表达式中的文字数,并通过核析取技术来实现因子分解和公共子表达式消去,进而实现对表

达为多项式的线性 DSP 转换和非线性滤波的优化。虽然该方法提供了一种系统的方式来优化多项式,但是该多项式表达不是正则的,因此严重缩小了优化的范围。

针对传统方法的不足,本文采用有序的、简化的和正则带权值广义表(Weighted Generalized List, WGL)^[11]建模多项式数据通路,然后按照一定的规则对该 WGL 进行分解,进而得到可调度的 DFG。

1 WGL 简介

对于多元整系数多项式 $f(x_0, x_1, \dots, x_{n-1})$, 变量 x_0 在 $x_0=0$ 时的泰勒展开结果为:

$$f(x_0, x_1, \dots, x_{n-1}) = f(0, x_1, \dots, x_{n-1}) + x_0 \frac{\partial}{\partial x_0} f(0, x_1, \dots, x_{n-1}) + \frac{x_0^2}{2!} \frac{\partial^2}{\partial x_0^2} f(0, x_1, \dots, x_{n-1}) + \dots + \frac{x_0^i}{i!} \frac{\partial^i}{\partial x_0^i} f(0, x_1, \dots, x_{n-1}) + \dots \quad (1)$$

其中, $\frac{\partial}{\partial x_0} f(0, x_1, \dots, x_{n-1})$, $\frac{\partial^2}{\partial x_0^2} f(0, x_1, \dots, x_{n-1})$ 和 $\frac{x_0^i}{i!} \frac{\partial^i}{\partial x_0^i} f(0, x_1, \dots, x_{n-1})$ 分别是多项式 f 在 $x_0=0$ 对 x_0 的一阶、二阶和高阶偏导。对于式(1)中的每一项按照同样的方法迭代地对其余的变量按照一定的顺序继续进行泰勒展开,其结果存储在广义表中(见图 1),该广义表称为 WGL^[11]。

本文受国家自然科学基金(61473047),中央高校基本科研业务费专项资金(310824161004)资助。

李东海(1977—),男,博士,讲师,主要研究方向为 VLSI 的高级综合与形式验证方法, E-mail: dhli@chd.edu.cn; 朱晓晨(1992—),男,硕士生,主要研究方向为嵌入式系统和云计算; 范中磊(1970—),男,副研究员,主要研究方向为嵌入式系统和云计算; 杨小军(1972—),男,教授,主要研究方向为无线传感器网络。

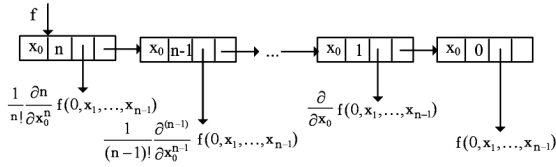


图1 WGL分解规则

图2所示为WGL表达 $F(A, B, C) = AB + AC$ ，其中变量顺序为 $\{A, B, C\}$ 。第一级分解与变量 A 相关，其中项 $F(A=0, B, C)=0$ ，图中未给出；第二项 $F'(A=0, B, C)=B+C$ ，用乘法边连接节点 B 进行表达，这表示项 $F'(A=0, B, C)=B+C$ 。第二级分解与变量 B 相关，其中项 $G(B=0, C)=C$ ，用加法边连接节点 C 进行表达，项 $G'(B=0, C)=1$ ，节点 B 的乘法项为1。第三级分解与变量 C 相关，其中 $C(0)=0, C'=1$ ，节点 C 的乘法项为1，最终构建的WGL如图2所示。

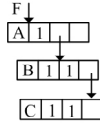


图2 WGL表达 $F(A, B, C) = AB + BC$

本文涉及到WGL表达非线性多项式时，将其转换成线性形式，即对于 $x^k (k \geq 1)$ ，采用线性的WGL进行表达时可以转化为 $x^k = x_1 x_2 \cdots x_k$ ，其中 $x_i = x_j$ 。下文中提到的WGL都为线性的WGL。

2 基于割的WGL分解

本节提出的WGL到DFG的转换方法是通过迭代地执行基于割的WGL分解，在介绍该分解过程之前首先介绍以下几个概念。

定义1 析取分解是把WGL分解成两个子WGL的和。

定义2 加法割施加于一个加法边，把WGL划分成两个子WGL，结果形成析取分解，即： $F = F_1 + F_2$ ，其中 F_1 和 F_2 为被分割的子WGL。

定义3 如果从WGL的根节点开始到终节点的所有路径都经过某节点，那么该节点称之为支配节点。

如图3(b)所示，节点 E 为支配节点。

定义4 连接分解把WGL分解成两个子WGL的积。

定义5 乘法割施加于一个支配节点，把WGL划分成两个子WGL，结果形成连接分解，即： $F = F_1 * F_2$ ，其中 F_1 和 F_2 为被分割的子WGL。

乘法割在WGL中用横条表示。图3(b)所示的乘法割 M_1 把WGL $F_2 = D(E+H)$ 分割成两个子WGL $F_{21} = D$ 和 $F_{22} = E+H$ 的积。

定义6 如果加法(乘法)割以加法(乘法)的方式把WGL划分成两个子表达，并且不增加最初WGL确定的运算个数，那么该加法(乘法)割称为可允许割，否则该割称为不允许割。

基于割的WGL分解的目的是找到一个可允许割序列。具体的分解算法如下所示，该算法的输入为一个将要分解的WGL，输出为一个可允许割序列SQU。其中，变量 i 和 j 的初始值均为0，序列SQU的初始化为空。

Procedure AdmissibleCut Sequence (WGL){

LV=WGL中自顶向下广度遍历WGL节点的列表(从右向左遍历);

while (LV $\neq \emptyset$) {

lv=LV中第一个可允许割节点;

if (节点lv是支配节点) then {

SQU=SQU $\cup \{M_i\}$;

$i=i+1$;

用乘法割把相应WGL分解成两个子sWGL; //连接分解

if (某个子sWGL中只包含1个节点lvs) then

LV=LV-{lvs};

}

if (节点lv的加法边是可允许割) {

SQU=SQU $\cup \{A_j\}$;

$j=j+1$;

用加法割把相应WGL分解成两个子sWGL; //析取分解

if (某个子sWGL中只包含1个节点lvs) then

LV=LV-{lvs};

}

return SQU;

}

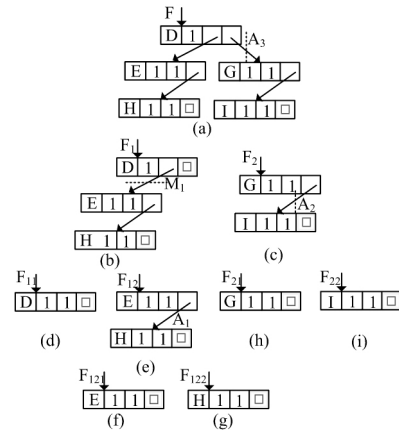


图3 WGL分解过程

该算法的具体过程为：首先自顶向下广度遍历WGL节点，并把相应的节点按照遍历的顺序存入列表LV中；然后迭代地从LV中找可允许割节点，若该节点是支配节点，那么可以进行连接分解，同时把相应的乘法割加入到可允许割序列中；若该节点的加法边是可允许割，那么可以进行析取分解，同时把相应的加法割加入到可允许割序列中。最终形成一个可允许割序列。如图3(a)所示，WGL表达多项式 $F = (G+D)(E+H)$ ，采用上面的算法，首先广度优先遍历该WGL可得 $LV = \{D, G, E, I, H\}$ ；对于节点 D ，可知其加法边为可允许割，则把加法割 A_3 加入SQU中，并把表达 $F = (G+D)(E+H)$ 的WGL分解为两个子WGL分别表达 $F_1 = D(E+H)$ 和 $F_2 = G+I$ ，如图3(b)和图3(c)所示；节点 D 在表达 $F_1 = D(E+H)$ 的子WGL中为支配节点，如图3(b)所示，则把乘法割 M_1 加入SQU中，并把表达 $F_1 = D(E+H)$ 的WGL分解为两个子WGL分别表达 $F_{11} = D$ 和 $F_{12} = E+H$ ，如图3(d)和图3(e)所示，由于 F_{11} 只包含节点 D ，则把节点 D 从LV中删除；对于节点 G ，可知其加法边为可允许割，如图3(c)所示，则把加法割 A_2 加入SQU中，并把表达 $F_2 = G+I$ 的WGL分解为两个子WGL分别表达 $F_{21} = G$ 和 $F_{22} = I$ ，如图3(h)和图3(i)所示，由于 F_{21} 和 F_{22} 分别只包含节点 G 和 I ，则把节点 G 和 I 从LV中删除；采用同样的方法对节点 E 进行操作，最终得到一个可允许割序列 $\{A_3, M_1, A_2, A_1\}$ 。同时由图3可知，加法(乘法)割的可允许性是动态属性，WGL(k)某个割在第 k 步可能是不允许的，但是WGL(j)某个子图该割在第 j 步($j > k$)会变成可允许的。在3图中，仅当应用割 A_3 后，割 M_1 是可允许的，随后应用割 M_1 后使得割 A_2 是可允

许的。找到一个可允许割序列是本算法的主要任务。

算法最终得到一个可允许割序列,然后反向遍历该可允许割序列,每访问该序列中的一个元素,相应的硬件运算(加法或乘法)就会被引进到 DFG 中去执行两个子表达的相应的运算。通过这种方式结合高级综合中的调度算法,一个可允许割序列最后会被转换成一个结构表达(由硬件运算组成),即数据流图。图 3 中的序列 (A_3, M_1, A_2, A_1) 最终被转换成如图 4 所示的数据流图。同时可得到如下引理。

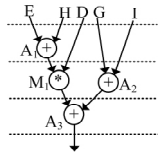


图 4 从图 3 中得到的数据流图

引理 每个可允许割序列产生一个唯一的 DFG。

证明:对于每个割序列,如果其是可允许的,会产生一个合法的 DFG。而且,对于给定的序列,不同的 DFG 意味着 DFG 节点(运算)的不同偏序,会得到不同的割序列,因此对于每个割序列仅仅会得到一个这样的 DFG。

3 实验结果

本文通过一些 DSP 设计来测试所提算法的有效性,如表 1 所列,其中 FIR(Finite Impulse Response)是有限脉冲响应滤波器,IIR(Infinite Impulse Response)是无限脉冲响应滤波器,Prodmat(Matrix Product Computation)是矩阵积计算,Elliptic 是椭圆滤波器,DCT(Discrete Cosine Transform)是离散余弦变换。针对 DSP 设计,首先采用文献[11]的方法建立其相应的 WGL,然后采用本文方法对相应的 WGL 进行分解,进而建立其对应的 DFG。表 1 给出了由 GUAT^[12]和本文算法分别对这些 DSP 设计进行综合的结果。由表 1 可知采用本文所提方法得到的 DFG 的控制步比 GUAT 少。

表 1 GUAT 和本文算法分别产生 DFG 的结果比较

设计	DFG:GUAT			DFG:本文算法		
	运算器数		延迟	运算器数		延迟
	加法	乘法		加法	乘法	
FIR16	15	16	16	15	8	5
IIR	4	4	5	4	4	4
Prodmat	48	64	4	48	64	3
Elliptic	20	0	11	36	0	7
DCT	48	64	4	48	64	3

同时针对 FIR16 和 IIR 滤波器,加上资源约束,然后对表 1 中得到的 DFG 进行重新综合,其结果如表 2 所列。

表 2 GUAT 和本文算法加上资源约束分别产生 DFG 的结果比较

设计	运算器数		延迟(ns)	
	加法	乘法	GUAT	本文算法
FIR16	1	1	330	210
	1	2	180	150
	1	4	170	150
	2	1	330	210
	2	2	180	130
	2	4	170	90
	4	1	330	210
	4	2	180	130
IIR	4	4	170	90
	1	1	90	40
	1	2	90	40
	2	1	60	30
	2	2	60	30

其中第 2 列和第 3 列分别为相应的资源约束数,即给定加法器和乘法器的个数,后两列分别为在资源的约束下采用 GUAT 和本文的算法分别对 FIR16 和 IIR 进行重新综合后得到的 DFG 的延迟(其中,假设加法器的延迟为 10ns,乘法器的延迟为 20ns,时钟周期设为 10ns)。由表 2 可知,采用本文方法得到的 DFG 的延迟比 GUAT 小。

结束语 本文基于 WGL 分解,实现了一种将初始算术规范转换成数据流的方法。该方法的任务是找到一个可允许割序列,根据可允许割序列进而得到一个可调度的 DFG。该 DFG 被综合后在延迟方面优于现存的方法。

参考文献

- [1] Del Barrio A A, Hermida J C R. A Distributed Clustered Architecture to Tackle Delay Variations in Datapath Synthesis [J]. IEEE Transaction on Computer-Aided Design of Integrated Circuits and Systems, 2016, 35(3): 419-432
- [2] Sierra R, Carreras C, Caffarena G. A Formal Method for Optimal High-Level Casting of Heterogeneous Fixed-Point Adders and Subtractors [J]. IEEE Transaction on Computer-Aided Design of Integrated Circuits and Systems, 2015, 34(1): 52-62
- [3] Jin S, Kim D, Nguyen T T, et al. Design and Implementation of a Pipelined Datapath for High-Speed Face Detection Using FPGA [J]. IEEE Transactions on Industrial Informatics, 2012, 8(1): 158-167
- [4] Zheng Hong-bin, Gurumani S T, Yang Li-wei, et al. High-Level Synthesis with Behavioral-Level Multicycle Path Analysis [J]. IEEE Transaction on Computer-Aided Design of Integrated Circuits and Systems, 2014, 33(12): 1832-1845
- [5] Gupta S, Savoiu N, Dutt N, et al. Using global code motion to improve the quality of results in high level synthesis [J]. IEEE Transaction on Computer-Aided Design of Integrated Circuits and Systems, 2004, 23(2): 302-312
- [6] Zilic Z, Karajica B. High-level Design of Integrated Microsystems-Arithmetic Perspective[C]// Proceedings of International Symposium on Robotic and Sensors Environments (ROSE), 2011. Montreal, Quebec, Canada, 2011: 77-82
- [7] Sjoval P, Virtanen J, Vanne J, et al. High-Level Synthesis Design Flow for HEVC Intra Encoder on SoC-FPGA [C]// Proceedings of Euromicro Conference on Digital System Design, 2015. Funchal, Madeira, Portugal, 2015: 49-56
- [8] Ghandali S, Alizadeh B, Navabi Z, et al. Polynomial Datapath Synthesis and Optimization Based on Vanishing Polynomial over Z_2^m and Algebraic Techniques[C]// Proceedings of International Conference on Formal Methods and Models for Codesign, 2012. Arlington, Virginia, 2012: 65-73
- [9] Ghandali S, Alizadeh B, Fujita M, et al. RTL Datapath Optimization Using System-level Transformations[C]// Proceedings of International Symposium on Quality Electronic Design, 2014. Santa Clara, CA, USA, 2014: 309-316
- [10] Ghandali S, Alizadeh B, Fujita M, et al. Automatic High-Level Data-Flow Synthesis and Optimization of Polynomial Datapaths Using Functional Decomposition [J]. IEEE Transaction on Computers, 2015, 64(6): 1579-1593
- [11] 李东海, 马光胜, 胡靖. 高层次数据通路的等价性验证方法[J]. 哈尔滨工程大学学报, 2008, 29(6): 583-588
- [12] Andriamaisaina C, Coussy P, Casseau E, et al. High-Level Synthesis for Designing Multimode Architectures[J]. IEEE Transaction on Computer-Aided Design of Integrated Circuits and Systems, 2010, 29(11): 419-432