一种 LDPC 码双向图环路检测新算法

李 博, 王 钢, 杨洪娟, 魏 民

(哈尔滨工业大学 通信技术研究所, 哈尔滨 150001, libolibolibo9999@163.com)

摘 要: 对一种计算通信网节点间全部路由的逻辑代数化算法进行改进,提出一种 LDPC 码双向图环路检测新算法.算法遵循逻辑代数运算规则,由 LDPC 码校验矩阵构造双向图的邻接矩阵,并对邻接矩阵中行与行之间进行整合与删除运算以及部分元素的修正运算,对于码长为 n,信息位为 k 的 LDPC 码只需 $(n-1)\times k$ 次整合和删除运算及 k 次元素修正运算,就能得到该双向图中所有的环路. 通过算例详细说明算法的计算过程,并验证其正确性和合理性.

关键词: LDPC 码; 双向图; 环路; 逻辑代数

中图分类号: TN911 **文献标志码:** A **文章编号:** 0367-6234(2010)07-1051-05

New algorithm of detecting cycles in bipartite graphs of LDPC codes

LI Bo, WANG Gang, YANG Hong-juan, WEI Min

(Communication Research Center, Harbin Institute of Technology, Harbin 150001, China, libolibolibo9999@163.com)

Abstract: In this paper, a logic algebraic algorithm in computing entire routes among the nodes of a communication network is improved, and a new algorithm of detecting cycles in bipartite graphs of LDPC codes is proposed. The algorithm obeys the logic algebraic calculation rules. Through integrating and deleting rows of relative matrix and correcting some elements of the matrix, the new algorithm can detect all the cycles in the bipartite graphs of LDPC codes. When the code length is n and information bits number is k, the integrating and deleting algorithm needs $(n-1)\times k$ times and the correcting algorithm needs k times. The validity and rationality of the algorithm are demonstrated in detail. The calculation course of the algorithm is shown by an example and the correctness is validated.

Key words: LDPC codes; bipartite graphs; cycles; logic algebra

LDPC 码是近年来编码界的一项重要成果[1].1/2 码率的二元 LDPC 码在高斯白噪声信道下的性能距离山农极限仅 0.004 5 dB[2],是目前距离山农极限最近的信道编码. 对 LDPC 码性能起决定作用的是其校验矩阵的结构,由 LDPC 码的双向图描述[3]. 由于 LDPC 码译码采用迭代译码[4],其算法的关键是基于在节点间传递的信息是统计独立的,当 LDPC 码校验矩阵对应的双向图中有环存在时,某一节点发出的信息经过一个环长的传递会被传回自身,从而造成自身信息的叠加,破坏了统计独立的假设,影响译码的准确性[5]. 所以在构造 LDPC 码时应该尽量减少环的存在,或是让环尽量大. 对于已经构造好的 LDPC 码,长度为 4 的环可以由校验矩阵直观看出来,但是对于环长大于 4 的环,则必须用相关算法才能检测. 在研究 LDPC 码编码的过程中,发现目前采用的环路检测方法有基于图论的邻接矩阵算法和双向图变换方法. 对于码长为 n 的 LDPC 码,邻接矩阵算法要对 $n\times n$ 的邻接矩阵进行 n 次自乘,才可能检测出所有环路的存在,首先随着码长的增加,其计算量将会以指数形式增加;其次,由于病态路径的存在,用这种方法检测双向图的环路并不是精确的. 采用双向图变换方法检测环路,需要手动改变双向图的结构,对于码长较小的 LDPC

收稿日期: 2009-05-20.

作者简介: 李 博(1983—),男,博士研究生;
王 钢(1962—),男,教授,博士生导师.

码可以直观的得到环路个数、环长以及经过的路径,但是对于码长很长的 LDPC 码,双向图变换方法是不切实际的. 也有学者给出了统计 LDPC 码环数的方法[6],但是不能给出具体环路经过的路径. 在研究数据通信网和 LDPC 码编码的过程中,对一种计算通信网节点间全部路由的逻辑代数化算法[7]进行改进,研究出一种利用逻辑代数计算规则检测双向图环路的新算法,具有以下几个特点:第一,可以精确检测出双向图中的所有环路;第二,算法快速收敛;第三,算法检测出的环路符合图论的基本规则,即沿环一周不会两次经过同一节点或者同一条边;第四,算法采用全逻辑代数运算,便于编写计算机程序进行实现. 研究 LDPC 码相关文献资料,并没有采用类似算法进行双向图环路检测的先例.

1 采用逻辑代数算法检测 LDPC 码双向图环路的规定和计算规则

用逻辑代数变量代表某两个节点之间相连接的边,在 LDPC 码双向图中,信息节点只与校验节点相连,信息节点之间,校验节点之间是没有边连接的. 由 LDPC 的迭代译码算法可知,信息在边上的传递过程是双向的,所以双向图的边是一种无向链路,用一个逻辑代数变量表示即可,如编号为 i 的边赋予逻辑符号变量 L_i. 当两节点之间没有边相连时,规定用逻辑“0”表示. 另外,因为节点本身永远可以进行自身连接,规定节点本身的连接用逻辑“1”表示.

规定“与”逻辑关系用“·”表示,“或”逻辑关系用“+”表示. “与”逻辑又称为逻辑乘积,多个表示边的逻辑变量的逻辑乘积,如:$L_1 \cdot L_2 \cdot L_3 \cdot L_4$ 或者 $L_1L_2L_3L_4$,表示由这些边串联构成一条信息通路. “或”逻辑又称为逻辑和,并联的信息通路,在“与或”逻辑表达式中,有多少个逻辑乘积项求逻辑和,就表示有多少个信息通路. 最后确定的经过特定信息节点的环路表达式也是“与或”逻辑表达式,每个逻辑乘积项表示一个环路,有几个逻辑乘积项进行“或”运算,就有几条环路.

算法中的所有计算都遵循逻辑代数运算法则. 在检测双向图环路的每一步计算过程中,计算的结果都要把逻辑表达式展开为“与或”逻辑表达式,并且要求按照以下计算规则进行化简,保持最简“与或”形式,才能进行下一步计算. 用 L_i,L_j 表示任意不相同边,计算与化简应用到的计算规则[6] 如下:

$$\begin{cases} L_i + 0 = L_i, \\ L_i \cdot 0 = 0, \\ L_i + 1 = 1, \\ L_i \cdot 1 = L_i, \\ L_i + L_i = L_i, \\ L_i \cdot L_i = L_i, \\ L_i + L_i \cdot L_j = L_i \cdot (1 + L_j). \end{cases} \tag{1}$$

2 LDPC 码双向图环路检测的新算法

由图论的知识可以知道,双向图存在其对应的邻接矩阵. 新算法首先根据双向图对应的校验矩阵构造相应的邻接矩阵. 邻接矩阵构造好之后,假定双向图中某个信息节点为源节点,然后对邻接矩阵进行整合运算,逐行删除非源节点元素. 对于码长为 n 的 LDPC 码,双向图中共有 n 个节点,包括信息节点和校验节点,只需要进行 $n-1$ 次邻接矩阵的整合运算和一次表达式的修正,即可得到包含该信息节点的所有环路. 然后假设其他信息节点为源节点,重复运算. 当所有节点都作为源节点进行运算后,就得到该双向图的所有环路. 算法唯一涉及到的运算就是逻辑代数运算,下面详细介绍具体的计算步骤与方法.

2.1 双向图邻接矩阵的构造

设 LDPC 码的码长为 n,信息位为 k,则校验位长度为 $n-k$,校验矩阵 $\boldsymbol{H} = [h_{ij}]_{(n-k)\times n}$,$h_{ij}$ 的取值为 0 或 1,$i = 1,2,\cdots,n-k$;$j = 1,2,\cdots,n$. 算法需要建立 $n \times n$ 阶邻接矩阵 $\boldsymbol{M} = [m_{ij}]_{n\times n}$,$m_{ij}$ 表示矩阵中的元素,代表从节点 i 到节点 j 的连接关系,$i = 1,2,\cdots,n$;$j = 1,2,\cdots,n$. 对于 m_{ij},当 $i = j$ 时,表示节点本身,$m_{ii} = 1$;当节点 i 到 j 没有边相连时,$m_{ij} = 0$;当节点 i 到 j 有边 L_i 相连接时,$m_{ij} = L_i$.

把双向图中信息节点和校验节点依次编号为 $1,2,\cdots,n$. 设双向图中总的边数为 s,按照信息节点从左到右的顺序把边编号为 $L_1,L_2,\cdots,L_s$. 对校验矩阵进行改造,得到改造后的校验矩阵 $H' = [h'_{ij}]_{(n-k)\times n}$,当 $h_{ij} = 0$ 时,$h'_{ij} = 0$;当 $h_{ij} = 1$ 时,$h'_{ij} = L_x(x = 1,2,\cdots,s)$,$L_x$ 为连接节点 i 和节点 j 的边. 构造双向图的邻接矩阵,如下式表示:

$$\boldsymbol{M} = \begin{bmatrix} \boldsymbol{I}_{k\times k} & \boldsymbol{H}'^{\mathrm{T}} \\ \boldsymbol{H}' & \boldsymbol{I}_{(n-k)\times(n-k)} \end{bmatrix}. \tag{2}$$

其中 $\boldsymbol{I}$ 为单位阵.

2.2 邻接矩阵的整合与删除运算

为了得到经过一个信息节点的所有环,首先

需要计算出以该信息节点为源节点,其他所有节点为目的节点的全部路由,采用邻接矩阵的整合与删除运算[7-8]. 一次整合与删除运算所采取的方法和遵循的规则为:若以信息节点 t 为源节点,如果要删除第 $k(k \neq t)$ 行,整合第 $i(i \neq k)$ 行,则应该以第 k 行为参考,按照图1箭头所示的顺序进行整合运算. 从画圆圈的元素出发,图中首尾相连的两个箭头为一个整合单元组,斜向下箭头表示“与”运算,正上箭头表示“或”运算. 图 1 中向下的大箭头表示以 k 行为参考对 i 行进行整合后的第 i 行的结果. 以两个虚线箭头所示的整合单元组为例,整合后的元素为

$$m'_{i2} = m_{ik} \cdot m_{k2} + m_{i2}.$$

由此可得到一次整合运算的公式为

$$m'_{ij} = m_{ik} \cdot m_{kj} + m_{ij}. \tag{3}$$

式中:m'_{ij} 为整合后矩阵中新元素的表达式,m_{ik}, m_{kj}, m_{ij} 为进行整合运算前邻接矩阵中的元素.

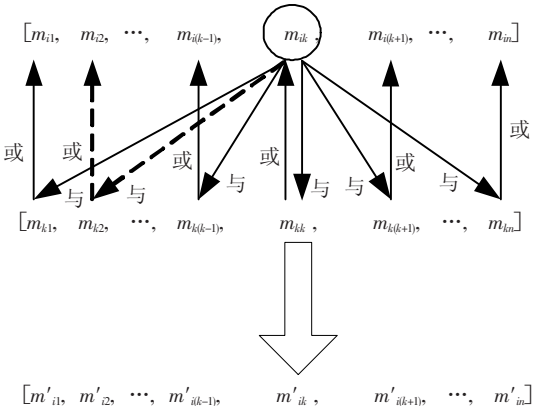

图 1　邻接矩阵的一次整合删除过程

当第 k 行与其他行逐一完成整合运算时,删除第 k 行,实现此次整合,即完成一次整合运算的邻接矩阵将减少一行,得到矩阵 $\boldsymbol{M}' = [m'_{ij}]_{(n-1)\times n}$. 按照这种方法,经过 $n-1$ 次整合删除运算,把除第 1 行外的其他 $n-1$ 行全部删除,便得到一个 $1 \times n$ 的矩阵 $\boldsymbol{M}^{(n-1)} = [m_{1j}^{(n-1)}]_{1\times n}$,上标$(n-1)$表示经过 $n-1$ 次整合运算. 矩阵 $\boldsymbol{M}^{(n-1)}$ 中各元素都是最简的“与或”表达式,其物理意义为以信息节点 1 为源节点,以其他所有节点为目的节点的全部路由. 并且,整合与删除邻接矩阵中 $n-1$ 行的先后顺序可以任意,不会影响到最终的计算结果.

2.3　求得经过指定节点的所有环路

在 2.2 中,得到了源节点到其他所有节点的全部路由. 要得到经过该节点的所有环路,只需要对 2.2 节中得到的计算结果进行修正. 在经过 $n-1$ 次整合与删除运算得到的矩阵 $\boldsymbol{M}^{(n-1)} = [m_{1j}^{(n-1)}]_{1\times n}$ 中,每个矩阵元素都是最简的“与或”表达式,其中有些元素进行“或”运算的各项中包含只由一条边组成的项,不妨定义其为单边项,假设包含单边项的元素为

$$m_{ij}^{(n-1)} = L_aL_b\cdots L_c + L_dL_e\cdots L_f + \cdots + L_g.$$

其中 a,b,c,d,e,f,g 的取值范围为 $1,2,\cdots,s$. 选出包含单边项的元素,对这些元素进行修正即可得到环路,修正的方法如下:首先用这些元素和单边项进行“与”运算,即

$$m_{ij}^{(n-1)} \cdot L_g = L_aL_b\cdots L_cL_g + L_dL_e\cdots L_fL_g + \cdots + L_g. \tag{4}$$

结果中仍然含有单边项 L_g,再将单边项删除,即得到环经过节点 i 和节点 j 的所有环路. 设环 C_{ij} 为经过节点 i 和节点 j 的环,则

$$C_{ij} = m_{ij}^{(n-1)} \cdot L_g - L_g = L_aL_b\cdots L_cL_g + L_dL_e\cdots L_fL_g. \tag{5}$$

对矩阵 $\boldsymbol{M}^{(n-1)}$ 中所有包含单边项的元素进行修正,即可得到经过源节点的所有环路. 而依次把所有信息节点设为源节点,进行以上运算,就可得到该 LDPC 码双向图中的所有环路.

3　算法的正确性与合理性论证及算例

3.1　算法的正确性与合理性论证

首先按照式(2)由 LDPC 码的校验矩阵构造其邻接矩阵. 由图论的知识可以知道,邻接矩阵是 $n \times n$ 的方阵,并且第 i 行表示节点 i 与其他节点的连接关系,第 i 列表示的意义与第 i 行完全相同,邻接矩阵是主对称阵. 因此,所有节点组成网络的连接关系既在行中体现,又在列中体现. 而 LDPC 码的校验矩阵的列对应信息节点的连接关系,行对应校验节点的连接关系,要用其构造邻接矩阵,就要补全列中的校验节点,补全行中的信息节点,由节点的对应关系,可知应该把校验矩阵 $\boldsymbol{H}$ 放在邻接矩阵的左下角,$\boldsymbol{H}$ 的转置 $\boldsymbol{H}^{\mathrm{T}}$ 放在邻接矩阵的右上角. 由于 LDPC 码的特殊性质,信息节点之间是没有连接的,校验节点之间也没有,而节点自身必定是相连的,所以补全部分的矩阵应该用单位阵. 这样构造的矩阵与根据双向图直接写出的邻接矩阵是完全相同的,因此,可以得到结论,按照式(2)得到的邻接矩阵是正确的.

对邻接矩阵的整合与删除运算是按照图 1 所示的方式,根据式(3)逐步进行的,并且在每一步的运算的过程中都严格按照式(1)进行化简,使其始终保持最简“与或”表达式,因此运算得到的

结果是正确的. 由于这一算法已经被应用于通信网路由表的计算过程中[9-10],不仅保证了节点和边的全面性和完整性,而且能够避免死循环和病态链路,所以算法的合理性是可以得到保证的.

经过对邻接矩阵的整合与删除运算,得到了以一个信息节点为源节点,其他所有节点为目的节点的全部路由链路,但还没有得到环路. 考虑与源节点有直接连接关系的节点,即存在单边项的矩阵元素对应的节点,源节点到这些节点的路由已经得到,只要在通过直接连接的边回到源节点,即得到一个环路. 因此按式(4)对包含单边项的矩阵元素进行修正,得到的将会是由源节点出发又回到源节点的环路,但是,单边项本身自乘表示信息由源节点出发经过直接相连的边到该目的节点,又经过该边返回源节点,由图论的知识知道,这不是一个环,是一个病态链路,因此要按照式(5)对其再次修正,去掉单边项,就得到经过源节点与该目的节点的所有环路. 而对没有和源节点直接相连的节点,是不可能再经过一次信息传递就形成环路的. 所以,由式(4)和式(5)得到的经过该源节点的环路是正确的.

由于 LDPC 码的信息节点只与校验节点相连,因此包含单边项的矩阵元素对应的节点肯定是校验节点,并且这种节点的个数就是该信息节点的边数. 当所有信息节点都作为源节点进行各步运算后,即可得到该 LDPC 码双向图的所有环路,不必再以校验节点为源节点进行计算,这是由于信息节点只与校验节点直接相连,计算得到的每个环路必定包含相关校验节点,即校验节点参与的环路在前边已经通过信息节点的计算全部得到. 因此,由算法得到的双向图的环路是全面的,该算法是合理的.

3.2 算例

以规则(8,2,4)LDPC 码为例,其对应的双向图如图 2 所示. 其中,$x_1,x_2,\cdots,x_8$ 表示信息节点,Z_1,Z_2,Z_3,Z_4 表示校验节点. 对节点和边的编号如图 2 所示.

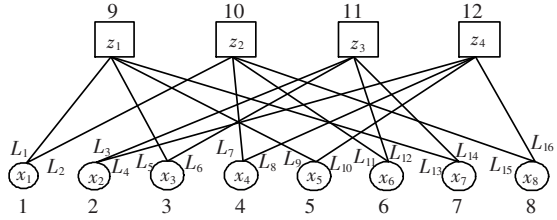

图 2 (8,4,2)LDPC 码双向图

该双向图对应的校验矩阵为

$$\boldsymbol{H}=\begin{bmatrix}1&0&1&0&1&0&1&0\\1&0&0&1&0&1&0&1\\0&1&1&0&0&1&1&0\\0&1&0&1&1&0&0&1\end{bmatrix}.$$

对校验矩阵进行改造,得到新的校验矩阵为

$$\boldsymbol{H}'=\begin{bmatrix}L_1&0&L_5&0&L_9&0&L_{13}&0\\L_2&0&0&L_7&0&L_{11}&0&L_{15}\\0&L_3&L_6&0&0&L_{12}&L_{14}&0\\0&L_4&0&L_8&L_{10}&0&0&L_{16}\end{bmatrix}.$$

按照式(2) 构造该双向图的邻接矩阵 $\boldsymbol{M}$,如式(6) 所示.

$$\boldsymbol{M}=\begin{bmatrix}\boldsymbol{A}&\boldsymbol{H}'^{\mathrm{T}}\\\boldsymbol{H}'&\boldsymbol{D}\end{bmatrix}. \tag{6}$$

其中

$$\boldsymbol{A}=\begin{bmatrix}1&&\\&\ddots&\\&&1\end{bmatrix}_{8\times8},$$

$$\boldsymbol{D}=\begin{bmatrix}1&&\\&\ddots&\\&&1\end{bmatrix}_{4\times4}.$$

按照图 1 及式(3) 的方法,对邻接矩阵 $\boldsymbol{M}$ 进行整合,设信息节点 1 为源节点. 依次整合删除其他的非源节点对应矩阵的行,经过 11 次整合与删除,最后得到一个 $1\times n$ 的矩阵 $\boldsymbol{M}^{(11)}$,表达式如下:

$$\boldsymbol{M}^{(11)}=\begin{bmatrix}m_{11}^{(11)}&m_{12}^{(11)}&m_{13}^{(11)}&m_{14}^{(11)}&m_{15}^{(11)}&m_{16}^{(11)}&m_{17}^{(11)}&m_{18}^{(11)}&m_{19}^{(11)}&m_{110}^{(11)}&m_{111}^{(11)}&m_{112}^{(11)}\end{bmatrix}.$$

其中各元素值为

$$m_{11}^{(11)}=1;$$

$$\begin{aligned}m_{12}^{(11)}=\;&L_1L_5L_6L_3+L_2L_{11}L_{12}L_3+L_1L_{13}L_{14}L_3+L_2L_7L_8L_4+L_1L_9L_{10}L_4+L_2L_{15}L_{16}L_4+L_2L_7L_8L_{10}L_9L_5L_6L_3+\\&L_2L_{15}L_{16}L_{10}L_9L_5L_6L_3+L_1L_9L_{10}L_8L_7L_{11}L_{12}L_3+L_1L_9L_{10}L_{16}L_{15}L_{11}L_{12}L_3+L_2L_7L_8L_{10}L_9L_{13}L_{14}L_3+\\&L_2L_{15}L_{16}L_{10}L_9L_{13}L_{14}L_3+L_1L_5L_6L_{12}L_{11}L_7L_8L_4+L_1L_{13}L_{14}L_{12}L_{11}L_7L_8L_4+L_2L_{11}L_{12}L_6L_5L_9L_{10}L_4+\\&L_2L_{11}L_{12}L_{14}L_{13}L_9L_{10}L_4+L_1L_5L_6L_{11}L_{12}L_{15}L_{16}L_4+L_1L_{13}L_{14}L_{12}L_{11}L_{15}L_{16}L_4;\end{aligned}$$

……

$$\begin{aligned}m_{19}^{(11)}=\;&L_2L_{11}L_{12}L_6L_5+L_2L_7L_8L_{10}L_9+L_2L_{15}L_{16}L_{10}L_9+L_2L_{11}L_{12}L_{14}L_{13}+L_2L_7L_8L_4L_3L_6L_5+L_2L_{15}L_{16}L_4L_3L_6L_5+\\&L_2L_{11}L_{12}L_3L_4L_{10}L_9+L_2L_7L_8L_4L_3L_{14}L_{13}+L_2L_{15}L_{16}L_4L_3L_{14}L_{13}+L_1;\end{aligned}$$

$$m_{110}^{(11)} = L_1L_9L_{10}L_8L_7 + L_1L_5L_6L_{12}L_{11} + L_1L_{13}L_{14}L_{12}L_{11} + L_1L_9L_{10}L_{16}L_{15} + L_1L_5L_6L_3L_4L_8L_7 + L_1L_{13}L_{14}L_3L_4L_8L_7 + L_1L_9L_{10}L_4L_3L_{12}L_{11} + L_1L_5L_6L_3L_4L_{16}L_{15} + L_1L_{13}L_{14}L_3L_4L_{16}L_{15} + L_2;$$

……

$$m_{112}^{(11)} = L_2L_7L_8 + L_1L_9L_{10} + L_2L_{15}L_{16} + L_1L_5L_6L_3L_4 + L_2L_{11}L_{12}L_3L_4 + L_1L_{13}L_{14}L_3L_4 + L_1L_5L_6L_{12}L_{11}L_7L_8 + L_1L_{13}L_{14}L_{12}L_{11}L_7L_8 + L_2L_{11}L_{12}L_6L_5L_9L_{10} + L_2L_{11}L_{12}L_{14}L_{13}L_9L_{10} + L_1L_5L_6L_{12}L_{11}L_{15}L_{16} + L_1L_{13}L_{14}L_{12}L_{11}L_{15}L_{16}$$

经过 11 次整合删除运算，得到的矩阵 $\boldsymbol{M}^{(11)}$ 中包含单边项的元素为 $m_{19}^{(11)}$ 和 $m_{110}^{(11)}$. 对它们按照式(4)和式(5)进行修正，得到经过节点 1 的环路，于是有

$$C_{19} = L_2L_{11}L_{12}L_6L_5L_1 + L_2L_7L_8L_{10}L_9L_1 + L_2L_{15}L_{16}L_{10}L_9 + L_2L_{11}L_{12}L_{14}L_{13}L_1 + L_2L_7L_8L_4L_3L_6L_5L_1 + L_2L_{15}L_{16}L_1L_4L_3L_6L_5L_1 + L_2L_{11}L_{12}L_3L_4L_{10}L_9L_1 + L_2L_7L_8L_4L_3L_{14}L_{13}L_1 + L_2L_{15}L_{16}L_4L_3L_{14}L_{13}\ L_1;$$

$$C_{110} = L_1L_9L_{10}L_8L_7L_2 + L_1L_5L_6L_{12}L_{11}L_2 + L_1L_{13}L_{14}L_{12}L_{11}L_2 + L_1L_9L_{10}L_{16}L_{15}L_2 + L_1L_5L_6L_3L_4L_8L_7L_2 + L_1L_{13}L_{14}L_3L_4L_8L_7L_2 + L_1L_9L_{10}L_4L_3L_{12}L_{11}L_2 + L_1L_5L_6L_3L_4L_{16}L_{15}L_2 + L_1L_{13}L_{14}L_3L_4L_{16}L_{15}\ L_2;$$

进而，把其他信息节点设为源节点，进行整合删除运算及修正运算，即可得到(8,4,2)LDPC 码对应的双向图中所有的环路. 其结果与其他方法计算出的环路完全相同，从而通过算例验证了算法的正确性.

4　结　论

对逻辑代数化路由算法进行改进，得出的 LDPC 码双向图的环路检测新算法，可以在构造 LDPC 码的过程中有效检测环路，然后按照消去环路的方法将其消去或部分消去，维持节点间信息传递的统计独立性，从而增加 LDPC 译码的准确性，提升 LDPC 码的性能. 该算法检测出的环路全面准确，符合图论的基本规定. 由于算法只涉及逻辑代数运算，而且规律性和目标性极强，收敛速度快，所以非常容易编写计算机程序，程序运行一次即可得出经过某个信息节点的全部环路，具有很强的实用性. 算法已通过程序调试，其正确性和可行性都得到了检验. 算法可以说是对图论方法的丰富与补充，它不仅适合检测 LDPC 码双向图中的环路，而且还可以用于各种类型网络拓扑图形的环路检测.

参考文献：

[1] GALLAGER R G. Low-density parity-check codes [J]. IRE Transactions on Information Theory, 1962, 8(1): 21 – 28.

[2] MACKAY D J C, NEAL R M. Near Shannon limit performance of low-density parity-check codes [J]. Electronics letters, 1996, 32(18):1645 – 1646.

[3] TANNER R M. A recursive approach to low complexity codes [J], IEEE Trans. Inform. Theory, 1981, IT – 27(5): 533 – 547.

[4] LUCAS R, FOSSORIER M, KOU Y, *et al.* Iterative decoding of one step majority logic decodable codes based on belief propagation [J]. IEEE Trans. Commun., 2000, 48(6): 931 – 937.

[5] SIPSER M, SPIELMAN D. Expander codes[J]. IEEE Transactions on Information Theory, 1996, 42(6): 1710 – 1722.

[6] 范俊，肖扬. 一种低密度奇偶校验码的环数统计方法[J]. 信号处理，2008，24(4)：635 – 639.

[7] DAI Fusheng, LIN Maoliu, QIAO Xiaolin, *et al.* A new source route algorithm of communication network[C]// IET International Conference on Wireless Mobile and Multimedia Networks Proceedings (ICWMMN 2006). Hangzhou: IET, 2006: 379 – 382.

[8] 戴伏生. 求解通信网节点间全部路由的逻辑代数化方法[J]. 数学的实践与认识，2006，36(2)：186 – 192.

[9] DAI Fusheng, SONG Lizhong, ZHANG Xiuzhen, Algebraic algorithm for reliability index weighted capacity of communication network [J]. Journal of Harbin Institute of Technology, 2003, 10(3): 290 – 294.

[10] 戴伏生，包学才，王小宇. 多约束路由的分层计算方法[J]. 南京邮电大学学报：自然科学版，2008，28(4)：38 – 43.

（编辑　张　宏）