基于符号EVBDD的安全多方计算

徐周波 俞强生 古天龙 宁黎华
(桂林电子科技大学广西可信软件重点实验室 桂林 541004)

摘 要 决策函数的有效表示是安全多方计算研究中的热点问题。符号描述技术是表示决策函数的一种新方法。针对基于代数决策图(ADD)的决策函数表示中出现的叶子节点规模膨胀以及导致协议面临的状态空间爆炸问题,引入边值二叉决策图(EVBDD)技术,给出了一种基于EVBDD的决策函数表示方法。该方法首先利用EVBDD结构,将决策函数描述为EVBDD的符号化形式,避免了传统ADD表示中出现的叶子节点规模膨胀现象。然后通过添加虚节点,解决了计算路径上出现的隐私泄露问题。在此基础上,提出了EVBDD的加解密算法,并设计了一种新的基于EVBDD的安全两方计算协议。最后,对协议的正确性、安全性和效率进行了分析。结果表明,与基于代数决策图的解决方案相比,新协议在效率上有明显的提高。

关键词 安全多方计算,决策函数,边值二叉决策图,状态空间爆炸
中图法分类号 TP309 **文献标识码** A **DOI** 10.11896/j.issn.1002-137X.2016.4.026

Secure Multi-party Computation Based on Symbolic Edge-valued Binary Decision Diagram

XU Zhou-bo YU Qiang-sheng GU Tian-long NING Li-hua
(Guangxi Key Laboratory of Trusted Software, Guilin University of Electronic Technology, Guilin 541004, China)

Abstract Efficient representation of the decision function is a hot problem in the study of secure multi-party computation. Symbol description technology is a novel method for the decision function representation. Aiming at the problem of leaf nodes expansion which leads to state explosion in the protocol produced by the algebraic decision diagram (ADD) representation of decision function, an edge-valued binary decision diagram (EVBDD) technique was introduced, and then a new method for decision function representation based on EVBDD was proposed. Firstly, the decision function is transformed to symbolic EVBDD form by using the structure of EVBDD, and thus the problem of leaf nodes expansion is avoided. Secondly, through adding virtual nodes, the privacy issues appeared on the computation path is solved. Furthermore, an EVBDD encryption algorithm was proposed, and a new secure two-party computation protocol based on EVBDD was designed. Finally, the correctness, security and efficiency of the new protocol was analyzed, demonstrating a considerable improvement in efficiency of the new protocol compared to the method based on ADD.

Keywords Secure multi-party computation, Decision function, Edge-valued binary decision diagram, State space explosion

1 引言

计算机和网络技术的迅猛发展为多方联合计算提供了大量的应用场景。然而,在信息化发展带来便利的同时,数据安全与隐私保护问题不断凸显,并成为了制约多方联合计算发展的最大障碍。安全多方计算(SMC)的核心思想就是在保护隐私的前提下,互不信任的参与方进行联合计算,并得到预期的执行结果。安全多方计算是信息安全领域的一个重要研究内容,也是现代密码学领域中的一个重要分支,在隐私保护基因计算[1]、数据信息比较[2]、保密计算[3]等多个领域有着广泛的应用。

安全多方计算的概念最早由图灵获得者Yao提出[4]。Yao通过"百万富翁"问题对安全计算进行了形象的阐述,即如何通过安全协议,在保护各自资产数量不为对方所知的前提下,计算出两个百万富翁谁更富有。决策函数的表示方式决定了其效率的高低,并得到了广泛研究,成为安全多方计算问题中的热点和难点问题。

在文献[4]的基础上,Yao提出了用布尔电路表示待计算的决策函数[5],Lindell于2006年对Yao的这种方法进行了详细论述和证明[6]。继Yao的开创性工作之后,许多的研究者对基于布尔电路技术的决策函数进行了优化。Pinkas和Schneider等人提出的混淆行简化游离异或的技术[7]结合了

到稿日期:2015-02-04 返修日期:2015-06-15 本文受国家自然科学基金(61100025,61262030,61363030),广西自然科学基金(2014GXNSFAA118354),广西高等学校高水平创新团队及卓越学者计划资助。
徐周波(1976-),女,博士,副教授,CCF高级会员,主要研究领域为符号计算、智能规划、约束求解,E-mail:xzbli_11@guet.edu.cn;俞强生(1987-),男,硕士生,主要研究领域为符号计算、安全多方计算,E-mail:844604194@qq.com;古天龙(1964-),男,博士,教授,博士生导师,主要研究领域为形式化方法、符号计算、知识工程,E-mail:cctlgu@guet.edu.cn;宁黎华(1981-),女,博士生,主要研究领域为智能计算、约束求解。

Kolesnikov 和 Schneider 提出的游离异或门技术[8]，以及 Naor 和 Pinkas 等提出的混淆行简化技术[9]，当电路门有超过33%以上为异或门时，其比基于秘密共享的技术有更好的通信效率。另外，Lindell、Pinkas 以及 Smart 提出将多个 2-输入的门合并为一个 d-输入的电路门，然后将异或门分离出来，d-输入的门再被分解为 2-输入的门[10]，能最大限度地减小非异或门的数量。除了以上减小电路大小的方法外，Jarvinen 等人提出用服务器发送给客户端一个防篡改的硬件令牌来代替服务器构建混淆函数，从而完全避免混淆函数的通信开销[11]；Iliev、Smith 提出利用加密协处理器来减小昂贵的加密计算开销[12]；Gunupudi、Goldwasser 和 Jarvinen 等人提出用可信任的硬件非交互式地一次性完成不经意传输协议[13-15]。同时 Malkhi 等人先后开发出了基于布尔电路的两方安全计算评估系统——FairPlay[16]、多方安全计算评估系统——FairPlayMP[17]，它们为安全多方计算走向实用奠定了基础。

由于布尔电路方法无法解决伪布尔函数问题，同时随着决策函数复杂性的增加，电路门规模会急剧增大，降低了协议效率。为了进一步提高决策函数的描述效率，在文献[5]的基础上，许多研究人员在如何进行安全多方计算形式化定义等方面进行了大量研究。2006 年，Kruger 等人设计了基于符号 OBDD 技术的决策函数表示[18]，首次提出了安全计算中决策函数的形式化表示，并通过实验数据证明了决策函数用 OBDD 表示时，协议的复杂度更低，效率更高。OBDD 是在布尔电路方法的基础上的一种形式化表示，因此在伪布尔函数中同样存在表示上的局限性。在文献[18]的基础上，古天龙等人提出基于 ADD 技术取代符号 OBDD 来刻画决策函数[19]，决策函数的表示形式由此得到了极大扩展。但同时也出现了另一个问题：叶子节点规模随着有限域中元素数目的增加会急剧膨胀，协议面临状态空间爆炸问题，其效率会受到较大的影响。

边值二叉决策图(Edge-Valued Binary Decision Diagram，EVBDD)作为表示伪布尔函数的一种新型数据结构，极大地改善了伪布尔函数和有限域取值函数的描述能力，在一定程度上对缓减叶子节点数目膨胀问题具有明显的效果。良好的隐私保护和对决策函数的有效表示是降低协议复杂度、提高协议效率的关键所在。鉴于此，本文引入边值二叉决策图(EVBDD)符号描述技术，利用其易操作和高紧凑性的优点，从决策函数的符号 EVBDD 表示出发，给出了基于 EVBDD 的加解密算法，并设计出了一个基于符号 EVBDD 的安全两方计算协议。

2 预备知识

2.1 安全多方计算模型

安全多方计算模型一般分为两种：①半诚实模型。如果所有参与者都是半诚实或诚实的，称此模型为半诚实模型。半诚实模型中的攻击者是被动的。②恶意模型。有恶意参与者的模型称为恶意模型，即攻击者能完全控制腐败方的模型。恶意模型中的攻击者是主动的。

在恶意模型中，参与者偏离了协议的宗旨，如果想要得到正确的计算结果，需要使用更多的密码协议及相关技术[20]。半诚实模型的概念最早由 Canetti[21] 提出，简单地说，半诚实模型是指协议的参与者完全按照协议规定的步骤执行，但他可能会保留协议执行过程中产生的中间结果，或者做出无法发现的违规动作。本文提出的协议在讨论安全性时，都是建立在半诚实模型的基础之上。

2.2 不经意传输协议

不经意传输协议是安全多方计算的一个重要工具，最早由 Rabin 提出，本文中使用的是二选一不经意传输协议(1-out-of-2)。通过以下定义可知，执行不经意传输协议可以让选择方使用其明文来从发送方安全获得与其明文对应的密文。

定义 1[22](二选一不经意传输协议) 参与者按行为划分为发送方和选择方。发送方输入一串 l-bit 的字符串对 $s_i^0, s_i^1 \in \{0,1\}^l$，其中 $i=1,2,\cdots,n$；选择方输入选择位 $b_i \in \{0,1\}$；协议执行完毕后，选择方获得字符串 $s_i^{b_i}$，但是无法知晓另一字符串 $s_i^{1-b_i}$，而发送方也无法获知选择方的选择 b_i。

2.3 同态加密(HE)

同态加密作为一种能对加密数据进行如同明文一样计算的加密工具，其本身可以作为完成安全评估协议的核心工具，但是考虑其本身相对较高的计算复杂度，安全评估协议的研究者往往使用其辅助混淆电路或混淆符号表示等技术来完成较复杂的函数或算法的安全评估。本文中使用的是一种语义安全的加同态公钥加密机制，加密机制的语义安全是指无法从特定密文中提取任何明文信息。

加同态加密算法具备如下性质：存在有效算法$\oplus$，$E(x+y)=E(x)\oplus E(y)$或者 $x+y=D(E(x)\oplus E(y))$成立，并且不泄漏 x 和 y。因此加同态算法也可以直接用于与常数 k 的乘法操作：$kE(x)=E(kx)$。本文中对同态加密的实例化使用的是文献[23]中的加密机制。

3 边值二叉决策图及决策函数的边值二叉决策图表示

3.1 边值二叉决策图

ADD 极大地改善了伪布尔函数和有限域取值函数的描述能力。但随着有限域中元素的增多，叶子节点数会增多，相应的节点数会增加很大。1994 年 Lai 等人[24]首次提出了能够紧凑描述伪布尔函数的边值二叉决策图，即在 ADD 的基础上，在边上引入权值来降低函数表示上的空间复杂度。

定义 2[25] 一个边值二叉决策图(EVBDD)是一个表示伪布尔函数 $f_i \in \{0,1\}^n \to Z$ 的二元组$\langle c, F\rangle$，其中 c 是一个整型常量，F 是一个具有根节点 r 的有向无环图$\langle V \cup T, E\rangle$，其中：

①V 是非叶子节点集，$r \in V$。T 是叶子节点集，该集合中有且仅有一个值为 0 的叶子节点，记作 0。E 是边集，其中每条边都有一个相应的权值，称该权值为边值。

②非叶子节点 v 具有四元组属性$\langle var, low, high, value\rangle$，其中 var 是节点 v 的标记变量，且 $var \in \{x_1,\cdots,x_n\}$，其中 $x_1,\cdots,x_n$ 均为布尔变量；low 和 $high$ 分别表示节点 v 的 0-分支节点和 1-分支节点，分别对应变量 var 取值为 0 和取值为 1 的情况；指向 low 和 $high$ 的有向边分别称作 0-边和 1-边。$value$ 是节点 v 的 1-边的边值，规定非叶子节点 v 的 0-边的边值为 0。

③对于一条指向根节点 r 的悬边，其边值为 c。

定义 3[24] 一个 EVBDD$\langle c,F\rangle$上的节点 u 表示了一个从$\{0,1\}^n$到 $\mathbf{Z}$ 的算术函数 $c+f$，其中 f 为根节点 r 所表示的函数。而每个 EVBDD$\langle c,F\rangle$上的节点 u 表示了一个从$\{0,1\}^n$到 $\mathbf{Z}$ 的算术函数 f^u，满足

①如果 u 是叶子节点，则 $f^u=0$；

②如果 u 是非叶子节点，则

$$f^u=u.var\cdot(value+f^{u.high})+(1-u.var)\cdot f^{u.low}$$
$$=u.var\cdot(value+f^u_{|u.var=1})+(1-u.var)\cdot f^u_{|u.var=0}$$

由定义 2 可见，在 EVBDD 中，一边值为 a 的边值指向一标记变量为 x_i 的节点 v，则节点 v 所表示的函数为：$\langle a,f_v\rangle=a+x_i\cdot f^v_{|x_i=1}+(1-x_i)\cdot f^v_{|x_i=0}$。

可见，EVBDD 是基于香农分解的。

定理 1[24] 对于从$\{0,1\}^n$到整数集 $\mathbf{Z}$ 的伪布尔函数 $f(x_1,\cdots,x_n)$，在给定的变量序下，存在伪布尔函数 $f(x_1,\cdots,x_n)$的唯一 EVBDD 表示。

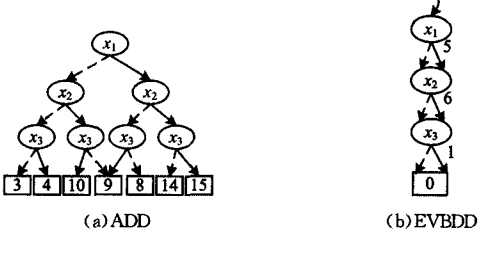
给定 EVBDD 中所有标记变量的一个变量序，如果在 EVBDD 中每一个从根节点到叶子节点的路径上，节点的标记变量均按该变量序出现，则称该 EVBDD 是有序的。本文决策函数的 EVBDD 表示是建立在有序的基础之上。

3.2 决策函数的边值二叉决策图表示

EVBDD 中的有限节点被分为叶子节点和内部节点两类，所有内部节点都有两个出度边，即 0-边和 1-边。根节点无入度边，叶子节点无出度边；变量的一组赋值决定于根节点到一个叶子节点的一条路径分支，该分支的各个节点所有标识的边值之和就是变量在这组赋值下所对应的函数值。同一函数在不同变量序下所得的 EVBDD 也是不同的。因此在求伪布尔函数的 EVBDD 时，需要说明变量的大小顺序。与 OBDD、ADD 的最大不同之处在于，EVBDD 所有边都附带有权值，并且只有一个叶子节点。

在决策函数的 EVBDD 表示中，约定 1-边用实线表示，边值列写在相应边旁，0-边用虚线表示，0-边上的边值为 0。在本文中，为降低节点密文规模，凡为 0 的边值在 EVBDD 的加密密文中均省略。其中，节点 v 取值后出度边上的边值用 $value(v)$表示。任一从根节点到叶子节点的路径上所对应变量取值下相应的函数是该路径上所有 1-边的边值的和。

例如，图 1(a)和图 1(b)分别是伪布尔函数 $f=3+5x_1+6x_2+x_3$ 在 ADD 和 EVBDD 符号表示下的图形结构，且变量序为 $x_1<x_2<x_3$。在输入模式(0,1,0)下，函数 f 所对应的函数值为 3+0+6+0=9。



(a)ADD　　(b)EVBDD

图 1 伪布尔函数 $f=3+5x_1+6x_2+x_3$ 的 ADD 和 EVBDD 表示

EVBDD 既能将决策函数拓展到伪布尔函数，又解决了随着有限域中元素增多，叶子节点数的膨胀问题。例如，利用 ADD 表示决策函数 $f(x_0,x_1,\cdots,x_{n-1})=2^{n-1}x_0+2^{n-2}x_1+\cdots+2^0x_{n-1}$，需要 $2^{n+1}-1$ 个节点；而用边值二叉决策图表示，则仅需 $n+1$ 个节点。在图 1 中，决策函数用 ADD 表示时需 15 个节点，而用 EVBDD 表示时仅需要 4 个。由此可见在 EVBDD 表示下，其描述效率更高。

4 边值二叉决策图的加密和解密

基于上述对决策函数的符号 EVBDD 表示，协议需要对 EVBDD 进行混淆加密，以及对密文正确解密。其中，对基于 EVBDD 的节点进行加密的算法如下。

算法 1 加密 EVBDD

输入：表示函数 f 的 EVBDD 有 n 个变量，变量序为 $x_1<x_2<\cdots<x_n$，节点规模为 d，其中有 d−1 个非终结节点 $P_i(1\leqslant i\leqslant d-1)$，1 个叶子节点 P_d，每个非终结节点对应一个值 level(P_i)，表示节点标注的变量在变量序中的位置，P_1 为根节点，且 level(P_1)=1。叶子节点作为算法结束标志，记为 end(v_d)，且 level(P_d)=n+1。

输出：密文$[\tilde{A}]=\{\langle\tilde{p}_1,\tilde{p}_2,\cdots,\tilde{p}_{n-k}\rangle\}$；n 对值 $\mathbf{W}_1,\mathbf{W}_2,\cdots,\mathbf{W}_n$。

1. choose a random permutation Π of the set $1,2,\cdots,n-k-1$ with Π[1]=1.
2. choose key $s_1=0^t$, random keys $s_i\in\{0,1\}^t$, $1<i\leqslant d-1$.
3. choose n pairs of random value $\mathbf{W}_k=\{\mathbf{w}_k^0=\langle s_k^0,\pi_k\rangle,\mathbf{w}_k^1=\langle s_k^1,1-\pi_k\rangle\}$, $1\leqslant k\leqslant n$, $s_k\in\{0,1\}^t$, $\pi_k\in\{0,1\}$.
4. for i=1 to d−1 do
5. let permuted index $\hat{l}=\Pi[i]$.
6. let 0 successor $i_0=low[i]$.
7. if $i_0\leqslant d-1$ then {p_{i_0} is a decision node}
8. let $\hat{l}_0=\Pi[i_0]$, $m^{\hat{l},0}=\langle$"decision", $\hat{l}_0,s_{\hat{l}_0}\rangle$.
9. else {p_{i_0} is a Leaf node}.
10. let $m^{\hat{l},0}=\langle$"classification", end(v_d)$\rangle$.
11. end if
12. let 1 successor $i_1=high[i]$.
13. if $i_1\leqslant d-1$ then
14. let $\hat{l}_1=\Pi[i_1]$, $m^{\hat{l},1}=\langle$"decision", $\hat{l}_1,s_{\hat{l}_1}\rangle$.
15. else let $m^{\hat{l},1}=\langle$"classification", end(v_d)$\rangle$.
16. end if
17. if $\tilde{p}_{\hat{l}}$ is a first node, let

$$\tilde{p}_{\hat{l}}=(\text{level}(\hat{l}),\text{value}(v_{init})),\ \text{Enc}^{\hat{l},\pi_{\text{level}(p_i)}}_{s_{\hat{l}}\oplus s_{\hat{l}}^{\pi_{\text{level}(p_i)}}}(\text{level}(\text{low}(\hat{l}))\ \|\ m^{\hat{l},\pi_{\text{level}(p_i)}}\ \|\ \text{value}(\text{level}(\text{low}(\hat{l}))),\ \text{Enc}^{\hat{l},1-\pi_{\text{level}(p_i)}}_{s_{\hat{l}}\oplus s_{\hat{l}}^{1-\pi_{\text{level}(p_i)}}}(\text{level}(\text{high}(\hat{l}))\ \|\ m^{\hat{l},1-\pi_{\text{level}(p_i)}}\ \|\ \text{value}(\text{high}(\hat{l})))).$$

18. else if $\tilde{p}_{\hat{l}}$ is not a first node, but a decision node, let $\tilde{p}_{\hat{l}}=(\text{level}(\hat{l}),\ \text{Enc}^{\hat{l},\pi_{\text{level}(p_i)}}_{s_{\hat{l}}\oplus s_{\hat{l}}^{\pi_{\text{level}(p_i)}}}(\text{level}(\text{low}(\hat{l}))\ \|\ m^{\hat{l},\pi_{\text{level}(p_i)}}\ \|\ \text{value}(\text{low}(\hat{l}))),\ \text{Enc}^{\hat{l},1-\pi_{\text{level}(p_i)}}_{s_{\hat{l}}\oplus s_{\hat{l}}^{1-\pi_{\text{level}(p_i)}}}(\text{level}(\text{high}(\hat{l}))\ \|\ m^{\hat{l},1-\pi_{\text{level}(p_i)}}\ \|\ \text{value}(\text{high}(\hat{l}))))$.
19. else, let $\tilde{p}_{\hat{l}}=(\text{level}(\hat{l}),\text{Enc}_{s_{\hat{l}}}(m^{\hat{l},\pi_{\text{level}(p_i)}}),\text{Enc}_{s_{\hat{l}}}(m^{\hat{l},1-\pi_{\text{level}(p_i)}}))$.

20. end if

21. end for

22. return $[\tilde{A}]=\{\langle \tilde{p}_1,\tilde{p}_2,\cdots,\tilde{p}_d\rangle\}$; $W_1,W_2,\cdots,W_n$

Bob 获得 Alice 发送的密文 EVBDD 和节点的解密密钥，通过 1-out-of-2 不经意传输获取的节点取值密钥，通过算法 2 进行解密，得到各个节点明文中的边值 c。本文关于 EVBDD 的解密算法如下。

算法 2 密文 EVBDD 的解密

输入：密文$[\tilde{A}]=\{\langle \tilde{p}_1,\tilde{p}_2,\cdots,\tilde{p}_d\rangle\}$；n 组值 $W_1,W_2,\cdots,W_n$。

输出：各个节点解密出的边值：$c=\langle value(\hat{l}_1),value(\hat{l}_2),\cdots,value(\hat{l}_n)\rangle$。

1. let $\hat{l}=1$; $S_{\hat{l}}=0^t$(start at root node)。

2. for i=1 to n do

3. let $\langle s_i,\pi_i\rangle=w_i$; $\langle c_{\hat{l}}^0,c_{\hat{l}}^1\rangle=\tilde{p}_i$.

4. if i=1, let $c_i=value(v_{init})$.

5. else if $i\neq 1$ && $i\neq n-k$, let $c_i=Dec_{s_{\hat{l}}\oplus s_{\hat{l}}}^{\hat{l},\pi_i}(c_{\hat{l}}^{\pi_i})$.

6. else, let $c_i=end(v_d)$.

7. end if

8. end for

9. let $C=\sum_{i=1}^{n}c_i$.

10. return C.

本文算法 1 和算法 2 是在基于 EVBDD 符号表示的基础上提出的，使用的语义安全的对称加密机制实例化为 $Enc_k^s(m)=m\oplus H(k\|s)=Dec_k^s(m)$，其中 $H(k\|s)$为安全散列算法 SHA256。

由上述加解密算法可知，计算的主要开销为安全散列算法 SHA256，并且在算法中 SHA256 的输入长度为 $t_0+\lceil\log_2 d\rceil$，t_0 为密钥长度，d 为决策函数规模。SHA256 算法的明文空间为 2^{64}-bit，其输入按 512-bit 分组进行处理，产生 256-bit 的摘要；在输入不足 448-bit 时，则无需分组且填充输入至 512-bit。在算法 1 中 SHA256 的密钥长度 t_0 取 80，并且在 SHA256 无需分组处理的前提下，算法 1 输入的节点规模最大数量可达到 2^{448-80}个。而在实际处理的决策函数规模不超过最大值时，算法中的 SHA256 的计算开销不变。因此，算法中的输入不影响 SHA256 的计算开销；从而算法 1 和算法 2 在输入的 EVBDD 有 n 个变量和 d 个节点数时，其计算开销为 $2d+n$ 次 SHA256 计算；其中密文规模为$2d(\lceil\log_2 d\rceil+t_0)$bits。关于加解密的更多细节分析可参考文献[23]和文献[19]。与 ADD 不同的是，EVBDD 叶子节点无需直接存储计算结果，其密文结构与内部节点一致，同样需要加解密。因此，d 不仅仅是内部节点，还包括叶子节点。

由本文的第 3 节可知，基于 EVBDD 技术的决策函数规模 d 比基于 ADD 的更低。因此，本文提出的 EVBDD 加解密算法比文献[19]的算法在效率上有较大提高：

(1)叶子节点仅作为协议结束标志，不存储计算结果，缩小了 EVBDD 的密文规模，尤其在有限域中元素数目较多，基于 ADD 出现叶子节点数目膨胀问题时；

(2)基于 EVBDD 的决策函数，其节点规模的降低，减少了协议中节点的加解密次数，节约了加解密时间。

5 基于 EVBDD 的安全两方计算协议

5.1 协议流程

在上述决策函数 EVBDD 表示以及加解密算法的基础上，给出基于符号 EVBDD 的安全两方计算协议流程。本文假设参与计算的双方为 Alice 和 Bob，协议具体内容如下。

输入：表示(伪)布尔函数 $f(x_1,x_2,\cdots,x_n)$的 EVBDD(f)，其中变量序为 $x_1<x_2<\cdots<x_n$。另外，Alice 的输入$x_a=(x_1,x_2,\cdots,x_k)$对应 EVBDD(f)中的前 k 个变量，Bob 的输入 x_b 对应后 $n-k$ 个变量。

输出：$C=f(x_a,x_b)$。

1. Alice 执行如下步骤：

步骤 1 用变量$(x_1,x_2,\cdots,x_k)$遍历 EVBDD(f)的前 k 个节点，约束后的初始节点记为：v_{init}。

步骤 2 为约束后的 EVBDD($full|_{x_a}$)随机产生 $n-k$ 对节点的取值密钥：$(s_1^0,s_1^1),(s_2^0,s_2^1),\cdots,(s_{n-k}^0,s_{n-k}^1)$，并为每个节点 v 分配节点密钥 s_v。

步骤 3 为 $k+1$ 层到 n 层的每个节点随机分配一个标签，用 $level(v)$表示节点 v。

步骤 4 为约束后的 EVBDD($full|_{x_a}$)填充虚节点。

步骤 5 加密 EVBDD($full|_{x_a}$)，每个节点密文都包含：代表该节点位置的节点标签、该节点不同取值下的两条子密文。每条子密文又包含：路径中下一节点的标签、下一节点的节点密钥、该节点取值时的边值(也称权值)。

步骤 6 Alice 将 $k+1$ 层到 n 层的节点密文和初始节点的节点密钥 $S_{v_{init}}$ 发送给 Bob。

2. Bob 执行如下步骤：

步骤 1 Bob 获取初始节点的节点密钥 $S_{v_{init}}$，并通过$n-k$ 次 1-out-of-2 不经意传输获得对应其输入的密钥 s_v^0,s_v^1。

步骤 2 Bob 用上述步骤获得密钥解密，解密从 Alice 步骤 5 获取的密文得到 C，即 $f(x_a,x_b)$。

节点密文结构：EVBDD 表示决策函数，协议的计算结果由计算路径中所有边值决定，节点的密文中要包含节点 v 的出度边的边值 $value(v)$，其中节点的密文结构如下：

$$(level(\hat{l}),Enc_{s_{\hat{l}}\oplus s_{\hat{l}}}^{\hat{l},\pi_{level(p_i)}}(level(low(\hat{l}))\|m^{\hat{l},\pi_{level(p_i)}}\|value(low(\hat{l}))),Enc_{s_{\hat{l}}\oplus s_{\hat{l}}}^{\hat{l},1-\pi_{level(p_i)}}(level(high(\hat{l}))\|m^{\hat{l},1-\pi_{level(p_i)}}\|value(high(\hat{l}))))$$

由于 EVBDD 的计算结果决定于边值，而不是存放在叶子节点，因此要对首节点和叶子节点的密文形式作特殊考虑：

(a)根据定义 2，首节点 v_{init} 计算之前存在一个入度边值：$value(v_{init})$。首节点的密文结构：

$$(level(\hat{l}),value(v_{init})Enc_{s_{\hat{l}}\oplus s_{\hat{l}}}^{\hat{l},\pi_{level(p_i)}}(level(low(\hat{l}))\|m^{\hat{l},\pi_{level(p_i)}}\|value(low(\hat{l}))),Enc_{s_{\hat{l}}\oplus s_{\hat{l}}}^{\hat{l},1-\pi_{level(p_i)}}(level(high(\hat{l}))\|m^{\hat{l},1-\pi_{level(p_i)}}\|value(high(\hat{l}))))$$

(b)文献[19]中的叶子节点用来存放计算结果，而根据定义 3，计算结果由所有边值共同确定，叶子节点处于计算路径的末端，不存放边值。因此，本文协议中的叶子节点仅仅作为

结束标志：$end(\hat{l})$，不存放与计算结果无关的信息；同时为了保证节点密文结构一致性，叶子节点的密文结构为：

$$(level(\hat{l}), Enc_{s_{\hat{l}}}^{\hat{l},\pi_{level(p_i)}}(end(\hat{l})), Enc_{s_{\hat{l}}}^{\hat{l},1-\pi_{level(p_i)}}(end(\hat{l})))$$

5.2 虚节点的添加

在EVBDD图中可能存在节点跳级的情况，这会导致约束EVBDD的参与者输入信息的泄露，图2(a)表示函数$f=x_1 \wedge x_2$，假设Alice的输入为x_1，Bob的输入为x_2。在Alice $x_1=1$的约束后，Bob需要经过节点x_2的计算才能到叶子节点。但如果Alice的约束$x_1=0$时，Bob可直接到达叶子节点。这种跳级现象的存在可能导致Bob从跳级现象中推出Alice的信息。对比图2(b)，在Alice加入约束之前，在图2(b)中添加虚节点，无论Alice的输入是0还是1，都必须让Bob先"计算"节点x_2才能到达叶子节点。当然，添加的节点不会出现在实际计算路径中。通过这种方法，可以有效地解决Alice信息泄露的问题。

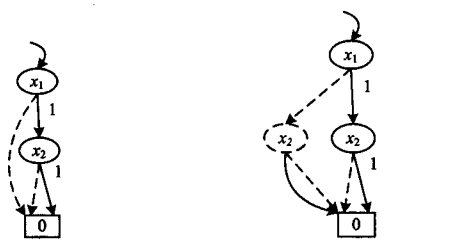


图2 EVBDD以及约束前添加虚节点

另一种可能导致信息泄露的情况就是在Alice不同的约束后，得到的EVBDD的规模不一样。例如函数$f(x_1,x_2,x_3,x_4)=(x_1 \wedge x_2) \vee (x_3 \wedge x_4)$，其符号描述如图3所示。

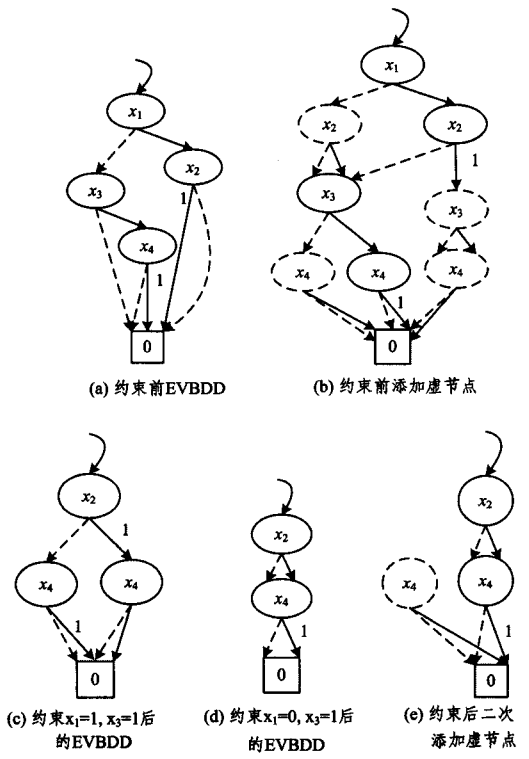


图3 EVBDD以及约束后添加虚节点

图3(a)为函数约束前未添加虚节点的EVBDD；图3(b)为Alice约束之前添加虚节点的EVBDD。虽然图3(b)解决了图3(a)中节点跳级的问题，但在图3(c)和图3(d)中，当Alice分别在不同约束$f|_{x_1=1,x_3=1}$和$f|_{x_1=0,x_3=1}$下，Bob依旧能从图的结构对比中推测出Alice的信息。针对这种情况，可以在Alice约束后二次添加虚节点，如图3(e)所示。

下面给出虚节点必须具备的性质：

(1)该节点内不含边值信息；

(2)该节点不会出现在实际计算路径中；

(3)该节点为非叶子节点。

性质(1)确保了所添加节点不会影响协议的复杂度。性质(2)保证了添加该类节点不会影响协议的执行效率。如果一个节点同时满足以上两种性质，则称该节点为虚节点。

6 协议的正确性、安全性和效率分析

6.1 正确性分析

下面分析Alice构造加密EVBDD后，Bob是如何正确地解密并最终得到正确的计算结果。假设Alice成功构造决策函数的EVBDD结构，在约束$x_a=(x_1,x_2,\cdots,x_k)$下，添加相应虚节点得到：$f|_{x_a}$。Bob无法得知约束前的Alice输入x_a以及约束后的EVBDD结构。Alice为约束后EVBDD结构中的各个节点分配一个节点密钥s_v，并连同解密后的节点密文一起发送给Bob。同时，Alice根据Bob的可能取值为所有非叶子节点分配一对取值密钥$(s_v^{\pi}, s_v^{1-\pi})$，假设此时$\pi=0$。若Bob得到加密后的节点为p_v，其密文中的两条子密文为：c_1、c_2。Bob根据此节点取值π，经过不经意传输协议在Alice对应该节点的取值密钥对$(s_v^{\pi}, s_v^{1-\pi})$中获得其中一个取值密钥s_v^{π}。Bob根据s_v和s_v^{π}可解出两条子密文中的一条，即得到计算路径中节点p_v的下一个节点的位置以及加密后的新的两条子密文。重复上述步骤即可到达叶子节点，得到正确的计算结果。因此，协议是正确的。

6.2 安全性分析

首先，参与协议的双方Alice和Bob均能按协议要求安全地输入各自的数据并严格地执行协议的各项步骤，这是由协议的执行环境(半诚实模型)所决定的。

其次，假设Alice提供节点密钥$s_v(v=1,2,\cdots,n-k)$和节点的取值密钥$(s_1^0,s_1^1),(s_2^0,s_2^1),\cdots,(s_{n-k}^0,s_{n-k}^1)$。Bob首先获取相应节点密钥，然后通过1-out-of-2不经意传输协议获取唯一的取值密钥，即只能获取s_v^0和s_v^1两者中的一个，同时Alice无法得知Bob的选择信息，这就确保了计算双方的隐私都能得到保护。Bob用获取的节点密钥和该节点的取值密钥可解密出该节点在相应取值后的下一节点，而且是唯一的。以此类推，能得到唯一的一条路径安全地到达叶子节点。由此可见，在这条唯一计算路径的求解过程中，参与者的信息都得到了保护。

最后，在协议执行之前，Alice加入了约束，隐藏了自身的EVBDD结构，加上不经意传输协议选择时的唯一性，保障了Alice的信息安全。由此可知，协议是安全的。

6.3 效率分析

本文主要从密文空间入手对协议效率进行比较分析，因为密文空间的大小直接影响到通信的开销。而决策函数的规模决定了密文空间的大小，是影响加解密时间的一大因素。决策函数规模越小，密文规模越小，加解密时间越短，协议效率越高。

两方安全计算中较为常见的是数据保密计算问题，因此，本文协议1在效率分析时，通过两位 t-bits 整数的与运算，与文献[19]基于 ADD 表示时的规模以及 SHA256 执行的次数的角度进行比较，如表1所列。

表1 基于 ADD、EVBDD 的密文规模和 SHA256 执行次数的比较

function		and-4	and-8	and-16
ADD-based[19]	size(nodes)	44	522	79024
	execute SHA256 (times)	96	1060	158080
	size(bits)	7568	93960	15330656
EVBDD-based	size(nodes)	29	267	13489
	execute SHA256 (times)	66	550	27010
	size(bits)	4930	47526	2535932

and-t：两整数按位与运算，Alice 和 Bob 各持有一个 t-bits 整数的输入(共计 $2t$ 个变量)，输出为一个 t-bits 整数。

随着有限域中元素的增多，决策函数在不同表示形式下，其叶子节点数的变化也会不同。为此，设定 $t=4,8,16$。其中 ADD-based、EVBDD-based 为决策函数的两种不同符号表示。

从表1的数据对比可知，随着有限域中元素数目的不断增长，基于 ADD 描述决策函数已不能很好满足协议的效率需求。而基于 EVBDD 的密文规模相对更小。

以上针对实例进行了分析。不失一般性，对任意函数，在最坏情况下，EVBDD 的叶子规模为1，总规模为 2^{2t}；而 ADD 的叶子规模为 2^t，总规模达到 $2^{2t}+2^t-1$。因此，基于 EVBDD 的决策函数的规模更小，协议效率更高。

结束语 本文针对决策函数基于符号 ADD 技术时叶子节点数膨胀以及协议面临的状态空间爆炸问题，引入符号 EVBDD 刻画决策函数，给出了协议基于 EVBDD 的加解密算法，并设计了一个基于 EVBDD 的安全两方计算协议。随后，证明了协议在不同函数下基于不同符号表示时，基于 EVBDD 技术的决策函数的复杂度最低。最后，通过协议的正确性、安全性和效率分析说明了协议的有效性。与文献[19]相比，所提算法降低了决策函数的复杂度，提高了协议的效率。

本文提出的基于 EVBDD 的安全协议有效地缓解了决策函数中叶子节点数过大以及引起的状态空间爆炸问题。由于边的权值因子的出现，协议中节点的存储变得相对更加复杂。因此，基于 EVBDD 安全协议的优化将是下一步研究的方向。此外，本文协议仅仅从两方安全计算来分析问题，在下一步工作中，也可以对协议的多方安全计算问题进行研究。

参 考 文 献

[1] Katz J, Malka L. Secure text processing with applications to private DNA matching[C]// Proceedings of the 17th ACM Conference on Computer and Communications Security. ACM, 2010: 485-492

[2] Liu Wen, Wang Yong-bin. Secure Multi-Party Comparing protocol and Its Applications[J]. Acta Electronica Sinica, 2012, 40(5): 871-876(in Chinese)
刘文，王永滨. 安全多方信息比较相等协议及其应用[J]. 电子学报，2012，40(5)：871-876

[3] Li Shun-dong, Wang Dao-shun. Efficient Secure Multiparty Computation Based on Homomorphic Encryption[J]. Acta Electronica Sinica, 2013, 41(4): 798-803(in Chinese)
李顺东，王道顺. 基于同态加密的高效多方保密计算[J]. 电子学报，2013，41(4)：798-803

[4] Yao A C. Protocols for secure computations[C]// 2013 IEEE 54th Annual Symposium on Foundations of Computer Science. IEEE, 1982: 160-164

[5] Yao A C. How to generate and exchange secrets[C]//27th Annual Symposium on Foundations of Computer Science, 1986. IEEE, 1986: 162-167

[6] Lindell Y, Pinkas B. A proof of security of Yao's protocol for two-party computation[J]. Journal of Cryptology, 2009, 22(2): 161-188

[7] Pinkas B, Schneider T, Smart N P, et al. Secure two-party computation is practical [M] // Advances in Cryptology-ASIACRYPT 2009. Springer Berlin Heidelberg, 2009: 250-267

[8] Kolesnikov V, Schneider T. Improved garbled circuit: Free XOR gates and applications [M] // Automata, Languages and Programming. Springer Berlin Heidelberg, 2008: 486-498

[9] Naor M, Pinkas B, Sumner R. Privacy preserving auctions and mechanism design[C]// Proceedings of the 1st ACM Conference on Electronic Commerce. ACM, 1999: 129-139

[10] Lindell Y, Pinkas B, Smart N P. Implementing two-party computation efficiently with security against malicious adversaries [M]// Security and Cryptography for Networks. Springer Berlin Heidelberg, 2008: 2-20

[11] Jarvinen K, Kolesnikov V, Sadeghi A R, et al. Embedded SFE: Offloading server and network using hardware tokens[M]// Financial Cryptography and Data Security. Springer Berlin Heidelberg, 2010: 207-221

[12] Iliev A, Smith S W. Small, stupid, and scalable: secure computing with faerieplay[C]// Proceedings of the fifth ACM Workshop on Scalable Trusted Computing. ACM, 2010: 41-52

[13] Gunupudi V, Tate S R. Generalized non-interactive oblivious transfer using count-limited objects with applications to secure mobile agents[M]// Financial Cryptography and Data Security. Springer Berlin Heidelberg, 2008: 98-112

[14] Goldwasser S, Kalai Y T, Rothblum G N. One-time programs [M]// Advances in Cryptology-CRYPTO 2008. Springer Berlin Heidelberg, 2008: 39-56

[15] Jarvinen K, Kolesnikov V, Sadeghi A R, et al. Garbled circuits for leakage-resilience: Hardware implementation and evaluation of one-time programs[M]// Cryptographic Hardware and Embedded Systems, CHES 2010. Springer Berlin Heidelberg, 2010: 383-397

[16] Malkhi D, Nisan N, Pinkas B, et al. Fairplay-Secure Two-Party Computation System[C]// USENIX Security Symposium. 2004: 287-302

[17] Ben-David A, Nisan N, Pinkas B. FairplayMP: a system for secure multi-party computation [C] // Proceedings of the 15th ACM Conference on Computer and Communications Cecurity. ACM, 2008: 257-266

[18] Kruger L, Jha S, Goh E J, et al. Secure function evaluation with ordered binary decision diagrams[C]// Proceedings of the 13th ACM Conference on Computer and Communications Security. ACM, 2006: 410-420

[19] Gu Tian-long, He Zhong-chun, Chang Liang, et al. Secure Evaluation of Classification Algorithms Based on Symbolic ADD and Linear Multi-Branching Program[J]. Acta Electronica Sinica,

2014,42(5):940-947(in Chinese)
古天龙,何仲春,常亮,等.基于符号 ADD 和线性多分支程序的分类算法安全评估[J].电子学报,2014,42(5):940-947
[20] Goldreich O. Foundations of cryptography [M]. Beijing: Publishing house of electronics industry,2003(in Chinese)
Goldreich O.密码学基础:英文版[M].北京:电子工业出版社,2003
[21] Canetti R. Studies in secure multiparty computation and applications[D]. The Weizmann Institute of Science,1996
[22] Rabin M O. How To Exchange Secrets with Oblivious Transfer [J]. IACR Cryptology ePrint Archive,2005:187
[23] Paillier P. Public-key cryptosystems based on composite degree residuosity classes [C] // Advances in Cryptology—EUROCRYPT'99. Springer Berlin Heidelberg,1999:223-238
[24] Lai Y T,Pedram M,Vrudhula S B K. EVBDD-based algorithms for integer linear programming, spectral transformation, and function decomposition [J]. IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems,1994,13(8):959-975
[25] Gu Tian-long,Xu Zhou-bo. Ordered binary decision diagram and its application[M]. Beijing: Science Press Ltd.,2009:17-57(in Chinese)
古天龙,徐周波.有序二叉树决策图及应用[M].北京:科学出版社,2009:17-57

(上接第 105 页)

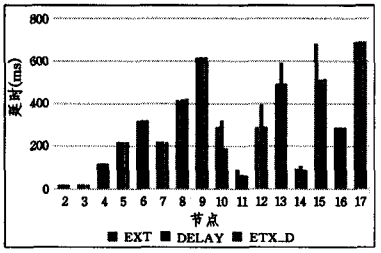


图 14 有损链路状态下的延时

从图 13 中可以看出,理想链路状态下,由于 ETX 并没有考虑延时的原因,并且形成的网络拓扑是比较稀疏的,图 7 中 10 号节点选择了唤醒周期较长的 11 号节点作为父亲节点,13 号节点选择了 12 号节点作为父亲节点,并没有同图 6 和图 8 中的拓扑一样去选择唤醒周期较小的父亲节点:3 号和 8 号节点,因此在采用 ETX 作为路由度量形成的网络拓扑中,10 号节点成功发送数据包给 1 号 root 节点会比采用延时和改进算法 ETX_D(见图 6、图 8)作为路由度量时所用的时间长。

有损链路的状态下,对于 13 号节点,由于延时路由度量并没有考虑数据包收发成功率的因素,因此图 10 和图 6 中,13 号节点都选择 8 号节点作为父亲节点,而对于考虑到了数据包收发成功率因素(即最少发送次数)的 ETX 和 ETX_D,图 11 和图 12 中,13 号节点则选择了 6 号节点作为父亲节点。在有损链路中,由于存在丢包问题,因此成功发送一个数据包与发送该数据包的次数有关,较少的发送次数也会降低延时,故在采用 ETX 和 ETX_D 作为路由度量形成的网络拓扑(见图 11、图 12)中,13 号节点成功发送数据包给 1 号 root 节点会产生较小的延时,如图 14 中给出的 13 号节点的延时比较所示。对于 10 号节点,比较延时路由度量(见图 10)和 ETX_D 路由度量(见图 12),由于网络拥塞和丢包会影响延时,因此采用 ETX_D 的 10 号节点并没有选择 3 号节点作为父亲节点,而是选择子节点较少的 4 号节点作为父亲节点,比较 ETX 路由度量(见图 11)和 ETX_D 路由度量(见图 12),由于 10 号节点到 11 号节点的延时比 10 号节点到 4 号节点的延时长,并且 3 个节点间的 ETX 值也并不相同,因此图 12 中的 10 号节点会选择 4 号节点作为父亲节点,如图 14 中给出的 10 号节点在各情况下的延时比较,在采用 ETX_D 作为路由度量形成的网络拓扑中,10 号节点成功发送数据包给 1 号 root 节点所产生的延时最小。

结束语 本文介绍了一种改进的 ETX 算法,同时在cooja模拟器上给出了仿真的结果,并与普通延时和标准的 ETX 进行了比较。实验结果证明,改进的 ETX 算法更具有实际应用意义,在有损链路并且具有不同唤醒周期节点的 WSN 中,该算法可以使成功传输数据包所产生的延时最小。但也应该看到,本文给出的路由度量的计算比较简单,提出的影响延时的因素也比较少,更为详细和完善的设计方案还有待研究。

参 考 文 献

[1] Dunkels A, Gronvall B, Voigt T. Contiki-a Lightweight and Flexible Operating System for Tiny Networked Sensors[C]//Proceedings of the 29th Annual IEEE International Conference on Local Computer Networks. 2004
[2] Hu Qin-yan, Yin Chang-chuan. Research RPL routing protocol for wireless sensor networks[J]. Things of Technology, 2014(1):57-62(in Chinese)
胡芹艳,尹长川.无线传感网络中的 RPL 路由协议研[J].物联网技术,2014(1):57-62
[3] Winter T, Thubert P, Brandt A, et al. RPL: IPv6 Routing Protocol for Low-Power and Lossy Networks [S/OL]. RFC 6550 (Proposed Standard), Mar. 2012. http://www.ietf.org/rfc/rfc6550.txt
[4] Vasseur J P, Kim M, Pister K, et al. Routing Metrics Used for Path Calculation in Low-Power and Lossy Networks[OL]. http://www.ietf.org/rfc/rfcb551.txt
[5] De Couto D S J, Aguayo D, Bicket J, et al. A highthroughput path metric for multi-hop wireless routing[C]//Proceedings of the 9th ACM International Conference on Mobile Computing and Networking (MobiCom'03). San Diego, California, September 2003
[6] Osterlind F, Dunkels A, Eriksson J, et al. Crosslevel sensor network simulation with cooja[C]//Proceedings 2006 31st IEEE Conference on Local Computer Networks. Nov. 2006:641-648
[7] Thubert P. Objective Function Zero for the Routing Protocol for Low-Power and Lossy Networks (RPL) [S/OL]. RFC 6552 (Proposed Standard), Internet Engineering Task Force, Mar. 2012. http://www.ietf.org/rfc/rfc6552.txt
[8] Gnawali O, Levis P. The Minimum Rank with Hysteresis Objective Function[S/OL]RFC 6719 (Proposed Standard), Internet Engineering Task Force, Sep. 2012. http://www.ietf.org/rfc/rfc6719.txt
[9] Gonizzi P, Monica R, Ferrari G. Design and Evaluation of a Delay-Efficient RPL Routing Metric[C]//Department of Information Engineering University of Parma Parco Area delle Scienze 181/A Parma. Italy, July 2013:1573-1577