



(12)发明专利申请

(10)申请公布号 CN 109478996 A

(43)申请公布日 2019.03.15

(21)申请号 201780045098.4

(74)专利代理机构 永新专利商标代理有限公司
72002

(22)申请日 2017.07.17

代理人 王英 刘炳胜

(30)优先权数据

16180511.4 2016.07.21 EP

(51)Int.Cl.

H04L 9/00(2006.01)

(85)PCT国际申请进入国家阶段日

2019.01.21

(86)PCT国际申请的申请数据

PCT/EP2017/067966 2017.07.17

(87)PCT国际申请的公布数据

W02018/015325 EN 2018.01.25

(71)申请人 皇家飞利浦有限公司

地址 荷兰艾恩德霍芬

(72)发明人 H·J·J·H·施珀斯

P·M·H·M·A·戈里森 L·马林

权利要求书5页 说明书19页 附图4页

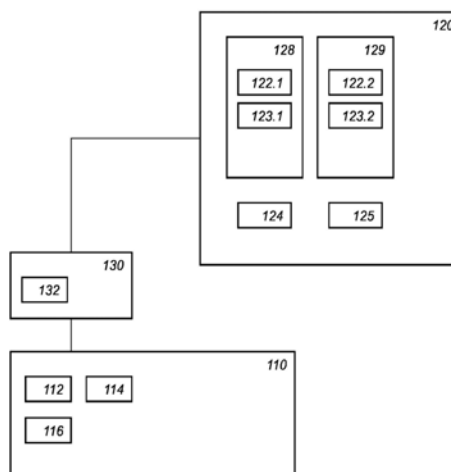
(54)发明名称

用于执行混淆算术的设备和方法

(57)摘要

一种计算设备(100)被布置为对域(F)的元素执行计算,域加法和域乘法是在所述域上定义的。编码的域元素是根据至少两种不同编码中的一种编码而编码的。计算管理器(130)被布置为通过以下操作选择性地将第一编码的域元素((a,b))和第二编码的域元素((c,d))相加或相乘:对于相加:将第二转换运算符单元应用于根据第二编码而编码的任何编码的域元素,并且应用加法运算符单元的集合;以及对于相乘:将第一转换运算符单元应用于根据第一编码而编码的任何编码的域元素,并且应用乘法运算符单元的集合。

100



1. 一种计算设备(100),其被布置为对域(F)的元素执行计算,域加法和域乘法是在所述域上定义的,所述计算设备包括:

-操作数存储(110),其被布置为存储编码的域元素(112、114、116;212),编码的域元素表示编码的形式的所述域的元素,其中,编码的域元素是根据至少两种不同的编码中的一种编码来编码的;

-所述至少两种不同的编码中的第一编码,其中,所述编码的域元素(x)被表示,

-所述至少两种不同的编码中的第二编码,其中,所述编码的域元素被表示为指数(s),所述编码的域元素是所述域的发生器(g)的所述指数的幂($x=g^s$),在所述操作数存储中的至少一些域元素是根据所述第一编码来编码的,并且至少一些域元素是根据所述第二编码来编码的,以隐藏对其执行的所述计算,

-运算符模块(120;220),其包括多个运算符单元,所述多个运算符单元包括:

-第一转换运算符单元,其被布置为:接收根据所述第一编码而编码的域元素,并且产生根据所述第二编码而重新编码的相同域元素,

-第二转换运算符单元,其被布置为:接收根据所述第二编码而编码的域元素,并且产生根据所述第一编码而重新编码的相同域元素,

-加法运算符单元的集合(128),所述加法运算符单元的集合被布置为将根据所述第一编码而编码的两个域元素相加,以及

-乘法运算符单元的集合(129),所述乘法运算符单元的集合被布置为通过将所述指数相加来将根据所述第二编码而编码的所述两个域元素相乘,

-计算管理器(130),其被布置为:

-接收第一编码的域元素和第二编码的域元素,所述第一编码的域元素和所述第二编码的域元素两者是根据来自所述操作数存储器110的所述第一编码或所述第二编码来编码的,

-通过以下操作选择性地将所述第一编码的域元素((a,b))和所述第二编码的域元素((c,d))相加或相乘:

-对于相加:将所述第二转换运算符单元应用于根据所述第二编码而编码的任何编码的域元素,并且应用所述加法运算符单元的集合,以及

-对于相乘:将所述第一转换运算符单元应用于根据所述第一编码而编码的任何编码的域元素,并且应用所述乘法运算符单元的集合。

2. 根据权利要求1所述的计算设备(100),其中:

-在所述第一编码中的所述编码的域元素被表示为两个元素的列表(213、215; (a,b)),编码的域元素是所述两个元素之间的差($x=a-b$),

-在所述第二编码中的所述编码的域元素被表示为两个元素((c,d))的列表,编码的域元素是所述域的发生器(g)的在所述两个元素之间的所述差的幂($x=g^{c-d}$)。

3. 根据前述权利要求中的任一项所述的计算设备,其中,所述加法运算符单元的集合和所述乘法运算符单元的集合包括以下中的至少一项:

-二元运算符单元(122;222、224),其被布置为:

-接收编码的域元素和参数,并且

-对所述编码的域元素和所述参数执行固定的计算,由此产生新的编码的域元素。

4. 根据前述权利要求中的任一项所述的计算设备, 其中:

- 所述第一转换运算符单元和所述第二转换运算符单元是一元的, 和/或
- 所述加法运算符单元的集合和所述乘法运算符单元的集合中的至少一个运算符单元是一元的,
- 一元运算符单元被布置为:
- 接收编码的域元素, 并且
- 对所述编码的域元素执行固定的计算, 由此产生新的编码的域元素。

5. 根据前述权利要求中的任一项所述的计算设备, 其中, 在所述加法运算符单元的集合与所述乘法运算符单元的集合之间定义第一一对一映射, 在被应用于相加的所述加法运算符单元的集合中的运算单元的序列通过所述第一映射被一对一地映射到在被应用于相乘的所述乘法运算符单元的集合中的运算单元的序列。

6. 根据前述权利要求中的任一项所述的计算设备, 其中, 在所述加法运算符单元的集合中的所述二元运算符的输入元素、输出元素和参数与在所述乘法运算符单元的集合中的所述二元运算符的输入元素、输出元素和参数之间定义第二一对一映射。

7. 根据权利要求2所述的计算设备, 其中:

- 所述加法运算符单元的集合和所述乘法运算符单元的集合包括二元向上运算符单元 (Δ , 222), 所述向上运算符单元被布置为:
- 接收根据所述第一编码或所述第二编码而编码的域元素 (x), 所述编码包括针对第一域元素 (a) 和第二域元素 (b) 的表示; 并且接收参数域元素 (c), 并且
- 计算第一域元素域乘以所述参数的倒数 (ac^{-1}) 并且计算第二域元素域乘以所述参数的倒数 (bc^{-1}), 所述新的编码的域元素 (y) 由所述两个计算的结果来表示 ($y = ac^{-1} - bc^{-1}$), 和/或

- 所述加法运算符单元的集合和所述乘法运算符单元的集合包括二元向下运算符单元 (∇ , 224), 所述向下运算符单元被布置为:

- 接收根据所述第一编码或所述第二编码而编码的域元素 (x), 所述编码包括针对第一域元素 (a) 和第二域元素 (b) 的表示; 并且接收参数域元素 (c), 并且
- 计算所述第一域元素域乘以所述参数 (ac) 并且计算所述第二域元素域乘以所述参数 (bc), 所述新的编码的域元素 (y) 由所述两个计算的结果来表示 ($y = ac - bc$)。

8. 根据权利要求2或7所述的计算设备, 其中, 所述加法运算符单元的集合和所述乘法运算符单元的集合包括一元框运算符单元 ($\square$, 226), 所述框运算符单元被布置为:

- 接收根据所述第一编码或所述第二编码而编码的域元素 (x), 所述编码包括针对第一域元素 (a) 和第二域元素 (b) 的表示; 并且接收参数域元素 (c), 并且
- 以编码的形式获得所述编码的域元素 (x) 加上固定增量的取反 ($-(x+1) = -x-1$)。

9. 根据权利要求7和8的组合的计算设备, 其中, 在需要时在转换之后第一编码的域元素 ((a, b)) 包括针对第一第一域元素 (a) 和第一第二域元素 (b) 的表示, 并且在需要时在转换之后所述第二编码的域元素 ((c, d)) 包括针对第二第一域元素 (c) 和第二第二域元素 (d) 的表示, 应用所述加法运算符单元的集合和应用所述乘法运算符单元的集合包括:

- 利用参数所述第二第一域元素 (c) 应用所述向上运算符单元,
- 所述框运算符单元,

- 利用参数所述第二第一域元素(c)应用所述向下运算符单元,
- 利用参数所述第二第二域元素(d)应用所述向上运算符单元,
- 所述框运算符单元,
- 利用参数所述第二第二域元素(d)应用所述向下运算符单元。

10. 根据前述权利要求中的任一项所述的计算设备,其中,所述运算符模块(120;220)包括存储装置,所述存储装置存储实施所述多个运算符单元的查找表。

11. 根据前述权利要求中的任一项所述的计算设备,其中,在第一编码中的两个域元素的所述列表被表示为指数 $((\alpha, \beta))$,所述两个域元素是所述域的公共基本元素(u)的所述指数增加到由所述指数指示的幂 $(x = u^\alpha - u^\beta)$ 。

12. 根据前述权利要求中的任一项所述的计算设备,其中,如果对应的指数(s)等于特殊元素 $(-\infty)$,则根据所述第二编码而编码的编码的域元素表示数字0。

13. 根据权利要求12所述的计算设备,其中:

-所述第二编码中的所述编码的域元素被表示为两个元素的列表 $((c, d))$,编码的域元素是所述域的发生器(g)的在所述两个元素之间的所述差的幂 $(x = g^{c-d})$,

-第二类型的编码的域元素中的所述元素(c,d)被限制为包括以下项的第二编码集合: $\mathbb{Z}_{n-1}$ 的一个或多个单元以及所述特殊元素,其中,n是所述域的元素的数量,并且其中,任选地

-第一类型的编码的域元素中的所述元素被限制为第一编码集合,其中,所述第一编码集合是所述域的差异集合,其中,所述第一编码集合和所述第二编码集合具有相同的大小。

14. 根据前述权利要求中的任一项所述的计算设备,其中,所述运算符模块(120;220)包括:

-第一虚拟转换运算符单元,其被布置为接收根据所述第一编码而编码的域元素,并且产生根据所述第一编码而重新编码的相同域元素,和/或

-第二虚拟转换运算符单元,其被布置为接收根据所述第二编码而编码的域元素,并且产生根据所述第二编码而重新编码的相同域元素。

15. 一种计算设备(100),其被布置为对域(F)的元素执行计算,域加法和域乘法是在所述域上定义的,所述计算设备包括:

-操作数存储(110),其被布置为存储编码的域元素(112、114、116;212),编码的域元素表示编码的形式的所述域的元素,其中,编码的域元素是根据至少两种不同的编码中的一种编码来编码的;

-所述至少两种不同的编码中的第一编码,其中,所述编码的域元素(x)被表示,

-所述至少两种不同的编码中的第二编码,其中,所述编码的域元素被表示为指数(s),所述编码的域元素是所述域的发生器(g)的指数的幂 $(x = g^s)$,在所述操作数存储中的至少一些域元素是根据所述第一编码来编码的,并且至少一些域元素是根据所述第二编码来编码的,以隐藏对其执行的所述计算,

-输入接口,其被布置为接收第一编码的域元素和第二编码的域元素,所述第一编码的域元素和所述第二编码的域元素两者是根据所述第一编码或所述第二编码来编码的,

-处理器电路,其被配置用于:

-多个运算符,所述多个运算符包括:

-第一转换运算符,其被布置为:接收根据所述第一编码而编码的域元素,并且产生根据所述第二编码而重新编码的相同域元素,

-第二转换运算符,其被布置为:接收根据所述第二编码而编码的域元素,并且产生根据所述第一编码而重新编码的相同域元素,

-加法运算符的集合(128),所述加法运算符的集合被布置为将根据所述第一编码而编码的两个域元素相加,以及

-乘法运算符的集合(129),所述乘法运算符的集合被布置为通过将所述指数相加来将根据所述第二编码而编码的所述两个域元素相乘,以及

-计算,所述计算包括:

-接收第一编码的域元素和第二编码的域元素,所述第一编码的域元素和所述第二编码的域元素两者是根据来自所述操作数存储器110的所述第一编码或所述第二编码来编码的,

-通过以下操作选择性地将所述第一编码的域元素 $((a, b))$ 和所述第二编码的域元素 $((c, d))$ 相加或相乘:

-对于相加:将所述第二转换运算符单元应用于根据所述第二编码而编码的任何编码的域元素,并且应用所述加法运算符单元的集合,以及

-对于相乘:将所述第一转换运算符单元应用于根据所述第一编码而编码的任何编码的域元素,并且应用所述乘法运算符单元的集合。

16.一种计算方法(300),其被布置为对域(F)的元素执行计算,域加法和域乘法是在所述域上定义的,计算设备包括:

-存储(310)编码的域元素(112、114、116;212),编码的域元素表示编码的形式的所述域的元素,其中,编码的域元素是根据至少两种不同的编码中的一种编码来编码的;

-所述至少两种不同的编码中的第一编码,其中,所述编码的域元素(x)被表示,

-所述至少两种不同的编码中的第二编码,其中,所述编码的域元素被表示为指数(s),所述编码的域元素是所述域的发生器(g)的所述指数的幂($x = g^s$),所存储的至少一些域元素是根据所述第一编码来编码的,并且至少一些域元素是根据所述第二编码来编码的,以隐藏对其执行的所述计算,

-接收(320)第一编码的域元素和第二编码的域元素,所述第一编码的域元素和所述第二编码的域元素两者是根据所述第一编码或所述第二编码来编码的,

-通过以下操作选择性地将所述第一编码的域元素 $((a, b))$ 和所述第二编码的域元素 $((c, d))$ 相加或相乘(330):

-对于相加:将第二转换运算符单元应用(342)于根据所述第二编码而编码的任何编码的域元素,并且应用(344)加法运算符单元的集合,所述第二转换运算符单元被布置为接收根据所述第二编码而编码的域元素,并且产生根据所述第一编码而重新编码的相同域元素,所述加法运算符的集合(128)被布置为将根据所述第一编码而编码的两个域元素相加,

-对于相乘:将第一转换运算符单元应用(352)于根据所述第一编码而编码的任何编码的域元素,并且应用(354)乘法运算符单元的集合,所述第一转换运算符单元被布置为接收根据所述第一编码而编码的域元素,并且产生根据所述第二编码而重新编码的相同域元素,所述乘法运算符的集合(129)被布置为通过将所述指数相加来将根据所述第二编码而

编码的所述两个域元素相乘。

17. 一种计算机可读介质 (1000), 其包括表示用于使处理器系统执行根据权利要求16所述的方法的指令的瞬态或非瞬态数据 (1020)。

用于执行混淆算术的设备和方法

技术领域

[0001] 本发明涉及一种计算设备、一种计算方法、一种计算机程序以及一种计算机可读介质。

背景技术

[0002] 在计算机中,针对各种任务执行计算。由于计算机是有限的,所以这些计算常常发生在有限域(field)中。域的范例是具有 p^n 个元素的伽罗瓦(Galois)域,其中, p 是素数。这样的域的常见范例是以素数取模的算术。针对计算机算术的另外的流行的域是具有 2^n 个元素的域。能够以各种方式来构造域,例如,在一些加密算法中使用通过以多项式取模的缩减而获得的域。

[0003] 在一些应用中,需要向攻击者隐藏关于程序的运行的信息。在所谓的白盒攻击模型中,假定攻击者具有对正在运行的计算机程序的详细访问权。即使在这种模型中,也希望尽可能地攻击者隐藏。具体地,使用密码术来向攻击者隐藏信息的敏感应用(诸如银行应用、内容保护等)在白盒模型中可能是易受攻击的。如果攻击者要读取例如被用于加密信息的秘密密钥,则攻击者可能能够自己对所述信息进行解密,由此获得财务、普通内容等。

[0004] 除了保护信息之外,更普遍地是还希望保护计算自身免受攻击者攻击。如果攻击者确切知道执行特定算法的位置,那么他可能能够将其攻击集中在程序中的正确地点;他可能能够对秘密算法进行逆向工程,例如专有加密或解密算法。

[0005] 使用当前的白盒技术很难保护一般的计算流程。例如,Chow等人的论文‘White-Box Cryptography and AES Implementation’展示了如何在白盒模型中保护一种特定算法(AES)。该技术可能无法直接被应用于保护通用计算机程序,亦即,在没有对程序的广泛人工分析的情况下不能够直接应用。例如,对Chow中所描述的类型表格或表格网络进行直接转换(例如,加法运算或乘法运算),将仍然允许攻击者仅通过观察访问哪个表格网络而推断何时执行加法或乘法。

发明内容

[0006] 根据权利要求提供了一种计算设备。在所述计算设备中,域元素可以以两种不同的方式来编码。在第一编码中,表示经编码的域元素(x),但是在第二编码中,经编码的域元素被表示为指数(s)。在这种情况下,经编码的域元素等于所述域的发生器(g)的指数的幂($x=g^s$)。

[0007] 这具有以下优点:可以对在第一编码中定义加法的操作进行镜像以定义乘法。这可以通过将域元素从第一编码转换为第二编码并且反之亦然的第一转换运算符单元和第二转换运算符单元来实现。

[0008] 换言之,即使使用允许表格网络的表示,例如,多个相互依赖的表格查找,也可以使用相同结构的表格网络进行加法以及乘法。例如,加法运算符的集合可以包括多个运算符,所述多个运算符被应用于输入元素或者运算符的先前结果的输出。

[0009] 方便的第一编码和第二编码是将元素 x 表示为两个元素 (a, b) 的列表。在第一编码中,所述元素是两个元素之间的差($x=a-b$)。在第二编码中,所述元素是发生器 g 的所述差的幂($x=g^{a-b}$)。该表示具有以下优点:可以使用仅接收3个域元素而不是4个域元素的运算符来定义加法,并且因此更小。此外,即使在第一编码内,乘法甚至也可以被表示为运算符的相似序列。因此,可以通过转换到第二编码类型或者通过保持在第一编码中而获得乘法;这扩大了针对混淆的选项。

[0010] 根据本发明的方法可以在计算机上被实施为计算机实施的方法、或者以专用硬件来实施、或者以这两者的组合来实施。针对根据本发明的方法的可执行代码可以被存储在计算机程序产品上。计算机程序产品的范例包括存储器设备、光学存储设备、集成电路、服务器、在线软件等。优选地,所述计算机程序产品包括被存储在计算机可读介质上的非瞬态程序代码,其用于在计算机上运行所述程序产品时执行根据本发明的方法。

[0011] 在优选实施例中,所述计算机程序包括计算机程序代码,当计算机程序在计算机上运行时,所述计算机程序代码适于执行根据本发明的方法的所有步骤。优选地,所述计算机程序被体现在计算机可读介质上。

[0012] 本发明的另一方面提供了一种使所述计算机程序可供下载的方法。当所述计算机程序被上传到例如Apple的App Store、Google的Play Store或Microsoft的Windows Store时,以及当所述计算机程序能从这样的商店下载时,使用该方面。

附图说明

[0013] 仅通过范例的方式,将参考附图来描述本发明的进一步细节、方面和实施例。为简单和清楚起见,图中的元件是说明性的,并且不一定按比例绘制。在附图中,与已经描述的元件相对应的元件可以具有相同的附图标记。在附图中:

[0014] 图1示意性示出了计算设备100的实施例的范例,

[0015] 图2a示意性示出了两个编码的域元素的实施例的范例,

[0016] 图2b示意性示出了运算符模块的实施例的范例,

[0017] 图3是示意性图示了计算的方法的流程图,

[0018] 图4a示意性示出了根据实施例的具有包括计算机程序的可写部分的计算机可读介质,

[0019] 图4b示意性示出了根据实施例的处理器系统的表示。

[0020] 图1-2b中的附图标记的列表:

[0021]	100	计算设备
[0022]	110	操作数 (operand) 存储
[0023]	112、114、116	编码的域元素
[0024]	120	运算符模块
[0025]	122.1、122.2	二元运算符单元
[0026]	123.1、123.2	一元运算符单元
[0027]	124	第一转换运算符单元
[0028]	125	第二转换运算符单元
[0029]	128	加法运算符单元的集合

[0030]	129	乘法运算符单元的集合
[0031]	130	计算管理器
[0032]	132	运算符的序列
[0033]	212	编码的域元素
[0034]	213、215	域元素
[0035]	232	编码的域元素
[0036]	233、235	域元素
[0037]	220	运算符模块
[0038]	222.1、222.2、 Δ	二元向上运算符单元
[0039]	224.1、224.2、 ∇	二元向下运算符单元
[0040]	226.1、226.2、 $\square$	一元框运算符单元
[0041]	228	加法运算符单元的集合
[0042]	229	乘法运算符单元的集合

具体实施方式

[0043] 尽管本发明可以具有许多不同形式的实施例,但是在附图中示出并且在本文中详细描述一个或多个具体实施例,应当理解,本公开被认为是本发明的原理的示例,而并非旨在将本发明限于所示和所描述的特定实施例。

[0044] 在下文中,为了便于理解,在操作中描述了实施例的各元件。然而,将显而易见的是,各个元件被布置为执行由其执行的所描述的功能。

[0045] 此外,本发明并不限于这些实施例,并且本发明在于上文所描述的或者在相互不同的从属权利要求中叙述的每个新颖特征或者各特征组合。

[0046] 图1示意性示出了计算设备100的实施例的范例。

[0047] 计算设备100被布置为对有限域F的元素执行计算。域是具有元素的有限集合的数学结构,其中,定义了域加法和域乘法。加法将被表示为+,乘法将被表示为 $\cdot$,或通过连接。元素的集合与加法一起形成阿贝尔(Abelian)群组,针对其的标识被表示为0。除了0之外的域是在乘法下的阿贝尔群组,例如,乘法是关联性的并且存在被表示为1的标识。F中的所有元素在加法下都有倒数,并且除了0之外的所有元素都在乘法下。加法和乘法是分布式的。

[0048] 本领域中已知许多不同的域。例如,以素数p取模的整数(例如, $\mathbb{Z}_p$)是域。例如,p可以是整数,诸如5、17等。如果f(x)是不可约多项式,则以数字和多项式取模的多项式的集合 $\mathbb{Z}_p[x]/f(x)$ 也形成域。在此,x被用作形式变量,象征性地用于表示所述域的元素中的一个元素。

[0049] 在各种域中的计算在计算机科学的许多分支中是重要的。例如,在二进制域中,所述域元素被写为二进制序列,并且加法对应于XOR运算。对于许多应用,重要的是在所述系统上对攻击者隐藏所述计算。例如,针对安全应用,可能希望所执行的确切计算保持隐藏,例如,可以从知道执行哪些计算来导出秘密密钥。如果攻击者在计算机软件代码正在被运行时可以完全访问所述计算机软件代码,则隐藏该信息尤其困难。尽管在本领域中已知不同的混淆技术,但是仍然希望进一步改进对域中的计算的混淆。

[0050] 其他算术,例如,以数M取模的算术,如模 2^{32} ,其在环 $\mathbb{Z}_{2^{32}}$ 中而不是在域 $GF(2^{32})$ 中,可以通过在更大的域中执行算术来建模。例如,通过选择素数 $p > 2M$,例如, $p = 2^{64} + 13$,并且在 $\mathbb{Z}_p$ 中执行算术并且每当操作数变得大于或等于M时减去M。

[0051] 计算设备100包括操作数存储110,操作数存储110被布置为存储编码的域元素。图1示出了编码的域元素112、114和116。存在对域元素进行编码的各种方式。编码的域元素以编码的形式来表示域的元素。下文讨论了一些编码。针对编码,存在解码映射,例如解密,其例如以一些常规的符号(例如,规范符号)将编码的域元素映射到普通的域元素;并且存在编码映射,其将普通的域元素映射到编码的域元素。给定的域元素可以以各种方式来编码,例如,编码的域元素不需要是独有的。在这样的情况下,所述解码是多对一映射。具有多种不同的编码提高了安全性,因为其使得更难以构建将编码的域元素映射到普通的域元素的表格。

[0052] 计算设备100使用至少两种不同的编码。第一编码可以是对域F的任何编码。例如,第一编码可以将域元素表示为域元素的列表。在实施例中,用于第一编码的有利编码是差编码,其中,域元素x被表示为一对域元素(a,b),从而域元素x是两个域元素之间的差($x = a - b$)。第二编码将域元素编码为发生器元素的指数。例如,第二编码可以是差指数编码,其中,域元素x被表示为一对域元素(a,b),从而域元素x是发生器的两个域元素之间的差的幂($x = g^{a-b}$)。

[0053] 图2a示意性示出了编码的域元素212的实施例的范例。编码的域元素212被编码为两个域元素213和215的列表。因此,以这种编码方式,相同的域被用于以编码的方式来表示其成员。在数学符号中,域元素x可以被编码为域元素的列表(a,b)。

[0054] 例如,在实施例中,所述编码的域元素是在所述两个域元素之间的差($x = a - b$)。

[0055] 可以进一步限制作为对(a,b)的表示。由于所述编码的域元素稍后将被用作对操作数单元的输入,通常被实施为查找表,因此减少域元素的可能表示的数量是有利的。这可以通过将元素(a,b)限制为F的所谓的差异集合来完成。差异集合是较大集合的子集,通常是域或环,使得较大集合的任何元素可以被表示为差异集合的元素的差。这在运算符单元被实施为表格时继而限制了运算符单元的大小。如果较大的集合包括0,则差异集合始终存在。

[0056] 在实施例中,所述列表的这两个元素是差异集合的元素。注意,对所述表示施加的附加约束可以被反映在表格运算中。例如,框运算符(参见下文)可以被表示为表格;如果使用差异表示,则可以以相同的方式来限制框运算符的输出,等等。

[0057] 通过要求列表表示的元素是差异集合的成员可以减少多少表示则取决于所需的安全性。如果存在元素的更多不同的表示,那么通过获知所述元素中的仅一个元素而获得的信息就更少。实际上,如果允许所有表示,则在对所述域元素的编码中的所述域元素中的一个域元素的知识不给出关于被编码的元素的信息。

[0058] 例如,为了将元素r编码为差异,可以选取该域的随机元素x,之后r可以被表示为例如 $r = (r-x) - x$,或者 $r = x - (x-r)$,或者作为元组((r-x,x)或(x,(x-r)))。可以通过枚举集合的各元素之间的所有差异并且对所述列表进行分类来完成对更受限制的表示的编码。

[0059] 通常,在编码的域表示中的域元素(例如,列表表示)不需要是普通的域元素,而是可以自身被编码以进一步混淆所述系统。因此,对程序进行逆向工程并且发现两个元素a和

b的攻击者不会立即知道差异 $a-b$ 。这种编码可以是域对自身的直接双射。以编码的元素作为输入的任何查找表都能够考虑双射,如在本领域中通常的那样。例如,可以定义从域到自身的双射E,在这种情况下,如果 $x=a-b$,则表示 (a,b) 可以作为Ea和Eb而被存储在操作数(operand)存储110中。也不需要将这两个数字彼此相邻地存储。

[0060] 编码E不需要是双射。例如,假设我们具体域 $A=\{0,1\}$ 以及范围 $B=\{0,1,2\}$ 。注意,B的尺寸大于A的尺寸。从A到B的关系E能够由 $\{(0,0), (1,1), (1,2)\}$ 给出,其中, (a,b) ,其中,a在A中并且b在B中应当被解读为“能够被映射到”。这引入了对来自A的元素1的加密的选择。关系E中需要的是其是可逆的。

[0061] 图2a还示出了编码的域元素232,类似于编码的域元素212,编码的域元素232包括两个域元素233和235。在数学符号中,域元素x可以被编码为域元素的列表 (a,b) 。

[0062] 针对第二编码,我们从观察开始,针对任何域,存在元素g,称为发生器,从而所述域的任何元素 $x \neq 0$ 能够被表示为针对特定i的 g^i 。如果域F的元素的数量被表示为n,则g的阶数为 $n-1$,亦即, $g^{n-1}=1$ 。因此,能够根据域F到 $\mathbb{Z}_{n-1}$ 来定义映射,其将x映射到i。在第二表示中,该域的任何非零元素能够由两个元素 (a,b) 来表示,使得 $i=a-b$,并且因此, $x=g^{a-b}$ 。这留下了如何表示数字0。这可以通过将一个特殊元素指定为负无穷大 $(-\infty)$ 来解决。例如,通过定义,该数字具有 $g^{-\infty}=0$ 的特性。第二编码可以被定义为:

[0063] 如果a和b两者不等于 $-\infty$,则 (a,b) 映射到 g^{a-b}

[0064] 如果a和b中的任何一个等于 $-\infty$,则 (a,b) 映射到0。

[0065] 集合 $\mathbb{Z}_{n-1} \cup \{-\infty\}$ 的基数等于域F的基数。

[0066] 在实施例中,在第二编码中所使用的元素可以是整数 $\{0, \dots, n\}$,其中,例如,整数 $\{0, \dots, n-1\}$ 是不等于 $-\infty$ 的元素,并且其中,整数n采取 $-\infty$ 的部分。然而,任何其他整数也可以被用作 $-\infty$ 。事实上,通过双射,任何数字都可以代表任何数字。

[0067] 同样地,在第二编码中,我们可以将元素a和b限制为 $\mathbb{Z}_{n-1} \cup \{-\infty\}$ 的特定子集。

[0068] 例如,在实施例中,两个双射被定义为从域F到整数0至 $n-1$ (包含)的 $E_{1,i}$,以及从 $\mathbb{Z}_{n-1} \cup \{-\infty\}$ 到整数0至 $n-1$ (包含)的两个双射 $E_{2,i}$ 。可以例如在编译时随机地选取两个双射。元素 $x=a-b$ 可以作为对 $(E_{1,2}(a), E_{1,2}(b))$ 而被存储在第一编码中,并且元素 $x=g^{a-b}$ 可以作为对 $(E_{2,2}(a), E_{2,2}(b))$ 而被存储在第二编码中,元素0可以通过将a和b中的任一项设置为负无穷大来表示。

[0069] 例如,考虑域GF(256),所述域具有256个元素。GF(256)的元素可以被认为是次数 ≤ 7 的x的多项式,其中,系数在GF(2)中。GF(2)仅仅是 $\{0,1\}$,具有二进制加法和乘法,但是没有进位:在GF(2)中 $1+1=0$ 。例如,AES加密使用以下缩减多项式来定义域: $x^8+x^4+x^3+x+1$ 。在这种情况下可以使用其他不可约多项式。通常,可以通过选取合适的不可约多项式来实施任何域。

[0070] 域的元素,特别是在AES中所使用的域GF(256),可以被认为是二进制多项式。总共有256个元素(因此,名称为GF(256))。

[0071] 在GF(256)中2个多项式的相加是简明的。这仅仅是多项式的正常加法,但是计算的系数发生在GF(2)中。后者可以被视为XOR操作。

[0072] 为了在GF(256)中乘以2个多项式,首先,像普通多项式一样乘以多项式但乘以两次,记住计算在GF(2)中进行。然后,将结果除以缩减多项式并且取余数。

[0073] 该域的元素可以被表示为二进制字符串,其中,每个比特表示形式变量 x 的指数的系数。实际上,这不是必需的,因为域与整数0到255之间的任何双射可以被用于表示域元素。在后一种情况下,可以使用查找表来执行域元素中的加法运算和乘法运算。

[0074] 域GF(256)具有许多发生器,例如,元素 $1+x$ 。我们将发生器元素称为 g 。注意, $g^{n-1}=1$,其中, n 是域的大小(在该范例中为256)。亦即, $(1+x)^{255}=1 \bmod x^8+x^4+x^3+x+1$ 。

[0075] 为了在第一编码中对元素进行编码,如使用具有形式变量 x 的表示的元素 $x^7+x^5+x^3+x$,或者当被表示为二进制字符串时的元素10101000,我们可以拾取随机元素 a ,如11011100,并且计算与10101000的差以获得01110100。由于该范例使用二进制域,因此这是XOR运算。因此,可以根据第一编码将10101000编码为(11011100,01110100);或者等效地编码为(220,116)。注意,在实际实施例中,可以使用不同的字符串,因为所述字符串自身也可以以编码的形式来存储。后一种编码可以是任何常规的编码,例如,随机双射等。需要将该级别上的编码与多个元素中的编码区分开。

[0076] 为了根据第二编码对元素 $x^7+x^5+x^3+x$ 进行编码,我们注意到, $x^7+x^5+x^3+x=(1+x)^{31} \bmod x^8+x^4+x^3+x+1$ 。注意,发生器 $1+x$ 是任意选择,其可以被隐含在运算符的实施中,并且不需要是显式的。在第二编码中,指数31是在0与254之间的数,并且可以被表示为两个整数之间的差,例如在0与254(包含)之间的两个整数。所述整数是模255(比域的阶少一)。例如,设置 $a=17$ 并且选择 b 使得 $a-b=31 \bmod 255$,从而 $b=241$ 。根据第二编码来编码的元素可以是(17,241)。为了表示元素0,我们利用元素负无穷大($-\infty$)来放大数字0到254。任何两个数字(其中的至少一个是无穷大)之间的差也是负无穷大。例如,让整数255表示附加数字负无穷大。我们定义 $g^{-\infty}=0$ 。针对0与254之间的任何 a ,数字0可以被表示为(255, a)、(a ,255)、(255,255)。同样地,针对第二编码,实施例不需要存储在编码下的实际数字,例如,在以上范例中为17、241和255,而是替代地,从整数0至255到其自身的任何双射可以被用于对第二编码中的数字进行编码。可以根据W0/2016/102445中所描述的系统(例如,其权利要求1)来完成对指数的编码。

[0077] 如与任何两种编码一样,根据第一编码而编码的元素可以例如使用查找表被转换成第二编码。如果可能有多个表示,则可以使用这种表示,例如,通过在查找表中具有一个或多个备选。在范例中,使用多个查找表,从而通过两个查找表以不同方式来转换元素。

[0078] 许多备选的第一/第二编码是可能的,其保留了由运算符单元的相似序列定义的乘法和加法的可能性。通常,元素 x 可以被表示为元素列表,例如,元素的向量 v ,从而 $l_1 \cdot v = x$ (第一编码)或 $l_2 \cdot v = s$,其中, $g^s = x$ (第二编码)。在此,所述编码由向量 l 定义。例如,可以使用 $l_1 = l_2 = (1, -1)$ 来获得差异编码。可以以相似的方式针对不同的 l 来定义向上/向下/框运算符,只要 l 映射到完整域,而不是映射到其子组。这可以通过在向量 l 中具有至少1个单位来实现。在实施例中, l 具有两个元素。

[0079] 在实施例中,编码中的两个域元素被表示为指数 $((\alpha, \beta))$ 。所述两个域元素是所述域的公共基本元素(u)的指数增加到由所述指数指示的幂($x = u^\alpha - u^\beta$)。这意味着编码的域元素 (α, β) 对应于普通的域元素 $x = u^\alpha - u^\beta$ 。这种类型的编码可以被称为日志形式。为了避免混淆,我们可以写 $(\alpha, \beta)_u$,如果我们想要明确表示使用了日志形式。同样地,在这种情况下,所述指数可以被限制为集合和/或被编码。

[0080] 返回到图1。操作数存储110可以包括常量。例如,在特定算法中所使用的常量域元

素,例如,在特定加密算法中,诸如加密、解密、MAC操作(消息认证码)、签名、签名验证等。操作数存储110还可以包括由用户输入的域元素,或者从例如在计算设备100外部的计算机接收到的域元素。计算设备100可以包括:编码单元,其用于将接收到的普通的域元素转换为编码的域元素;和/或解码单元,其用于例如在将编码的域元素发送到例如外部计算机之前将所述编码的域元素转换为普通的域元素。所述编码单元和所述解码单元可以仅限于一种类型的编码,例如,仅限于第一类型的编码。在这种情况下,转换表可以被用于将元素编码为第二类型以及从第二类型对元素进行编码。

[0081] 计算设备100还可以直接以编码的形式来接收外部域元素。在计算设备100外部,而不是在计算设备100的内部,可以使用另外的编码,例如加密。

[0082] 操作数存储228中的域元素中的一些域元素可以根据第一编码进行编码,而一些域元素可以根据第二编码进行编码。在实施例1中,在操作数存储228中的至少一些域元素根据第一编码被编码,并且至少一些域元素根据第二编码被编码。

[0083] 计算设备100包括运算符模块120。运算符模块120包括多个运算符单元。运算符单元可以被实施为单个查找表,或者被实施为多个查找表,例如,被实施为查找表网络。运算符单元也可以被实施为多条计算机指令,所述多条计算机指令被布置为执行所述运算符单元的功能。所述运算符单元可以被存储在计算设备100的电子存储器中。运算符模块120包括:第一转换运算符单元124,其被布置为接收根据第一编码而编码的域元素,并且被布置为产生根据第二编码而重新编码的相同域元素;以及第二转换运算符单元125,其被布置为接收根据第二编码而编码的域元素,并且被布置为产生根据第一编码而重新编码的相同域元素。

[0084] 此外,计算设备100包括加法运算符单元的集合128和乘法运算符单元的集合129。另外,在加法集合128中,两个运算符被示为:122.1和123.1。在乘法集合129中,示出了两个运算符:122.2和123.2。

[0085] 注意,不要将单位元素与设备的单元混淆。后者是执行特定功能的操作部分,前者是具有乘法逆的集合的元素。

[0086] 有趣的是,加法运算符单元的集合128被布置为将根据第一编码而编码的两个域元素相加。乘法运算符单元的集合129被布置为将在第二编码中根据第二编码而编码的两个域元素相加,从而在转换成第一编码之后其被相乘。这利用了对应关系 $g^x g^y = g^{x+y}$ 。注意,如果x和y中的一个为负无穷大,则该关系也成立,因为在这种情况下,x+y也是负无穷大,并且因此,等式的两边表示数字0。

[0087] 例如,考虑域元素x和y。以下操作是可能的:

[0088] -如果根据第一编码对两个域元素进行编码,则可以通过应用集合128来将其相加。

[0089] -如果根据第一编码对两个域元素中的一个或者这两个域元素进行编码,则可以通过将其转换为第二编码并且应用集合129来将其相乘。

[0090] -如果根据第二编码对两个域元素进行编码,则可以通过应用集合129来将其相乘。

[0091] -如果根据第二编码对两个域元素中的一个或者这两个域元素进行编码,则可以通过将其转换为第一编码并且应用集合128来将其相加。

[0092] 所述转换单元可以是一元运算符单元。然而,也能够使其接收参数,例如,在多个不同的表示当中进行选择。

[0093] 所述多个运算符单元中的至少一个运算符单元是二元运算符单元。图1示出了二元运算符单元122.1和122.2。运算符模块120还可以包含一个或多个一元运算符单元。图1示出了一元运算符单元123.1和123.2。

[0094] 诸如二元运算符单元122.1/2的二元运算符单元被布置为:

[0095] -接收编码的域元素和参数,并且

[0096] -对所述编码的域元素和所述参数执行固定的计算,由此产生新的编码的域元素。

[0097] 一元运算符单元(诸如一元运算符单元123.1/2)被布置为:

[0098] -接收编码的域元素,并且

[0099] -对所述编码的域元素执行固定的计算,由此产生新的编码的域元素。

[0100] 在二元和一元这两种情形中,所述编码的域元素可以例如通过计算管理器130(下文进一步讨论)从操作数存储器110接收。针对二元情况,也可以例如通过计算管理器130从操作数存储器110接收所述参数。

[0101] 由运算符单元执行的计算是固定的。如果向运算符提供不同的参数或编码的域元素,则其将执行计算的相同集合,尽管具有不同的输入。例如,运算符单元的计算可以包括(或者甚至由以下项组成)对参数或者其逆以及编码的表示的元素的域算术。例如,运算符单元的计算可以包括(或者甚至由以下项组成)域乘法和域加法。包括后者的域乘法可以是与固定域元素(例如,-1)的乘法。

[0102] 计算设备100包括计算管理器130。计算管理器130被布置为接收第一编码的域元素和第二编码的域元素。例如,计算管理器130可以被布置为从操作数存储器110取得第一编码的域元素和第二编码的域元素。例如,计算管理器130可以取得编码的域元素112和114。对所述域元素使用哪种类型的编码的信息可以被存储在计算设备100中,例如,被存储在操作数存储器110中。然而,能够将该信息隐含在例如处理流程中,例如,如由被布置为按需应用转换单元的计算管理器130执行的计算机程序中所定义的那样。

[0103] 计算设备100能够对第一编码的域元素和第二编码的域元素执行域加法和域乘法。选取哪种运算(加法还是乘法)取决于适用于计算设备100的应用。例如,可以执行加密操作,其需要执行大量的算术运算,包括乘法和加法两者。

[0104] 计算管理器130被布置为对根据第一编码而编码的元素执行域加法。计算管理器130被布置为使用至少从第二编码的域元素获得的参数将加法集合128中的多个运算符单元的序列应用于第一编码的域元素。例如,每个运算符单元可以以某种特定次序被应用于第一编码的域元素,其中的一些运算符单元可以被应用多次。例如,所述序列可以应用第一运算符,然后是第二运算符,然后是第三运算符,然后再次应用第三运算符,依此类推。

[0105] 类似地,计算管理器130被布置为对根据第二编码而编码的元素执行域加法(如果所述元素被转换为第一编码则实现乘法)。计算管理器130被布置为使用至少从第二编码的域元素获得的参数将乘法集合129中的多个运算符单元的序列应用于第一编码的域元素。

[0106] 有趣的是,计算管理器130可以用于在第一编码或第二编码中的加法的计算机指令可能不能与所应用的表的地址和内容区分开。具体地,被应用以在两个编码域中执行加法的表的数量可以相等。此外,参数的次序和来源可以相等。

[0107] 实际上,可以在加法运算符单元的集合与乘法运算符单元的集合之间定义第一一对一映射,被应用于相加的所述加法运算符单元的集合中的运算单元的序列通过第一映射被一对一地映射到被应用于相乘的所述乘法运算符单元的集合中的运算单元的序列。

[0108] 针对在二元运算符单元中所使用的参数可以进行相同的操作。例如,在所述加法运算符单元的集合中的二元运算符的输入元素、输出元素以及参数与在所述乘法运算符单元的集合中的二元运算符的输入元素、输出元素和参数之间定义第二一对一映射。

[0109] 计算管理器130可以被布置为根据第一编码直接对域元素执行域乘法。这甚至可以被布置为使得运算符的数量和次序与根据第一编码而编码的域元素的加法相同,然而,在这样的乘法中所使用的参数将是不同的。

[0110] 认为区分参数比区分表更容易。在白盒实施方案中,通常使用许多运算符单元。针对操作,可以定义多个运算符单元,例如,用于不同的编码。虚拟运算符单元可以不做任何事情或者仅改变编码等。然而,可以通过流分析来跟踪参数的不同使用,并且这样的分析可以是自动化的。此外,在实施例中,可以组合两种方案——隐藏表使用但是使用不同的参数流并且通过使用转换表来隐藏参数。

[0111] 例如,计算管理器130可以包括或者可以访问运算符132的序列。所述运算符的序列确定对哪些编码的域元素执行哪些运算符。序列132可以包括表示域加法的子序列以及表示域乘法的子序列。所述子序列可以在双射下映射到彼此。序列132还可以包括多个运算符的另外的应用,例如,转换运算符,或者甚至其他可能不相关的运算符。序列132可以被包括在程序中。序列132可以被存储在存储器中,例如,被存储在操作数存储110中。

[0112] 分析运算符模块120的操作的攻击者将看到运算的相同序列,而不管执行加法或乘法的事实。因此,攻击者不能够推断关于从对运算的分析中执行的计算的信息。以这种方式,计算设备的运行的重要方面对于攻击者是隐藏的。

[0113] 此外,在实施例中,可以将混淆技术应用于所述系统。具体地,多个运算可以作为查找表来执行。所述编码的域元素可以具有与普通的域元素的随机关系,例如,通过加密或随机编码。尽管攻击者可能能够看到应用了哪些表以及执行各种运算符单元的次序,但是他不能够从中推断出信息,因为域加法和域乘法的序列变得毫无意义。实际上,在这两种情况下都实际执行了加法。同时,参数变得不那么有用,因为其使用在这两种情况下都是相同的。

[0114] 一元运算特别适合用于实施为查找表,因为表的大小随着输入数量的增加而大大地增加。例如,计算设备100可以包括存储装置,所述存储装置存储实施框运算符单元的查找表(见下文)。

[0115] 除了域乘法和域加法之外,计算设备100可以被布置用于在域元素上的其他运算。例如,计算设备100可以包括存储装置,所述存储装置包括表格,所述表格表示不能够被(容易地)表示为加法和乘法的序列的运算。

[0116] 除了域加法和域乘法之外,还可以使用相同运算符的相同序列来表达其他运算。例如,在实施例中,计算管理器130被布置为通过使用至少从第二编码的域元素获得的参数将多个运算符单元的序列应用于第一编码的域元素来执行域减法,其中,针对域减法的序列减法与针对域加法的序列和针对域乘法的序列是相同的。例如,也可以用这种方式表示取反(negation)运算。

[0117] 在运算期间,可以例如如下地使用设备100。计算设备130从操作数存储装置110中取得第一编码的域元素和第二编码的域元素。在需要时,计算设备130应用第一转换或第二转换,并且从运算符模块120中选择运算符单元;例如,根据序列132并且使所选择的运算符单元被应用于第一编码的域元素。所述应用的中间结果可以被存储在运算符存储150中,例如,作为编码的中间域元素。计算设备130可以计算所需的参数。

[0118] 保护还可以使用硬件安全措施,但是尤其还可以是诸如混淆的软件保护。如果计算管理器130的操作数选择独立于参数计算,则混淆更有效。特别有效的软件保护包括白盒密码术的应用。

[0119] 在实施例中,操作数模块120包括:

[0120] -第一虚拟转换运算符单元,其被布置为接收根据第一编码而编码的域元素,并且被布置为产生根据第一编码而重新编码的相同域元素,和/或

[0121] -第二虚拟转换运算符单元,其被布置为接收根据第二编码而编码的域元素,并且被布置为产生根据第二编码而重新编码的相同域元素。

[0122] 例如,所述虚拟转换可以将输入域元素映射到相同编码类型中的相同域元素的不同表示。例如,控制模块130可以被配置为每次要执行加法或乘法时应用转换表,例如,如果不需要实际转换则应用虚拟转换表。以这种方式,针对加法和乘法的序列甚至更加相等。

[0123] 在实施例中,计算设备100包括参数单元,所述参数单元被布置为计算至少从第二编码的域元素获得的参数,以用于由计算管理器执行域乘法和域加法。例如,计算管理器130和参数单元可以被实施为有区别并且不同的电路。这样的计算系统可以在地理上分布,其中,计算管理器100和参数单元可以在地理上彼此分离。例如,地理间隔可能超过特定的所需距离,例如超过10km等。

[0124] 计算设备100可以被体现在虚拟机中。在这种情况下,序列132可以是在虚拟机上运行的程序的部分。常规虚拟机 (VM) 提供基本运算,其中有加法、减法、乘法、互斥等。这样的目的是双重的:虚拟机提供运算和原语,这些运算和原语不是由底层平台提供的,并且实现了在存储器受限的环境中特别有用的紧凑指令格式。

[0125] 然而,VM的指令中的许多指令仍然是基本的指令,如加法、减法、乘法等,其被直接映射到底层平台的指令上。由于这些通常被很好地理解,因此通过分析功耗和故障注入能够很容易地攻击VM。更重要的是,由于这种映射,很容易向个体操作中添加跟踪指令。

[0126] 根据实施例的虚拟机缓解了这种情况。虚拟机具有用于向上、向下和框操作的指令,例如,作为VM的指令集的部分。所述指令可能仅在特殊操作模式下可用,例如,安全模式。

[0127] 这些操作的序列与正在执行的实际功能之间的关系不那么直接。实际上,执行算术的代码可能如下:

[0128] 1. (任选地) 改变模式 (例如,通过分配一些地址), 然后:

[0129] 2. 实施算术的向上、向下和框指令的序列, 并且

[0130] 3. (任选地) 再次以模式改变结束。

[0131] 给定代数域的数量以及可能表示的变化,每个虚拟机都可以有自己的算法。

[0132] 所述虚拟机可以区分向上、向下和框指令的执行以及索引(参数)的计算。例如,这些活动可能被拆分。

[0133] 例如,一种活动可以是操作的序列的(预)获取,另一活动可以是指令的结果的实际查找。这些活动的途径实际上可能是实际的执行模型。

[0134] 可以使用表来实施向上、向下和框操作,这也有助于编码的实施方案。表驱动方案甚至可以在存储器受限的环境中使用,因为通过对底层域R的正确选取,新定义的算法仅需要小的表格。

[0135] 可以组合连续的向上和向下指令,例如, $\Delta_r \Delta_s = \Delta_{sr}$, $\nabla_r \nabla_s = \nabla_{sr}$, $\Delta_r = \nabla_{r-1}$,等。任意两个连续的向上或向下指令能够使用这些规则重写为单个向上或向下指令。计算设备或虚拟机的实施例包括组合单元。所述组合单元可以与计算管理器130协作,并且在执行由计算管理器130规划的运算符之前组合这些运算符。

[0136] 根据以上规则中的一项规则,所述组合单元组合两个连续的向上、向下或框操作并且将其组合成单个新的操作。然后,组合单元相应地组合对应的参数。这种类型的组合具有以下优点:参数不再与单个操作相对应,而是与组合的操作相对应。组合单元也可以是独立设备,例如,用于混淆给定序列。所述组合单元可以与生成运算符的序列的编译器集成。

[0137] 图2b示意性示出了运算符模块220的实施例的范例。该实施例包括加法运算符单元的集合228和乘法运算符单元的集合229。每个集合228、229包括三个运算符单元:两个二元运算器单元222.1/2和224.1/2以及一个一元运算器单元226.1/2。当在不同的集合中引用相同类型的运算符单元时,我们使用索引.1或.2来表示其在集合228或229中。我们可以省略索引,或者使用索引.1/2以指示该语句适用于集合228和集合229两者的操作数。

[0138] 实施例可以使用更多或更少的运算符单元。下文我们将首先假设域元素x根据第一编码被编码,作为两个域元素的列表(a,b),使得 $x=a-b$ 。在实施例中,域F具有至少4个或者至少8个元素等。尽管实施例中的表示可能是复杂的和/或被另外编码的,但是我们将不在下文包括该内容,以避免混淆讨论。

[0139] 我们将在运算符模块220中的运算符单元222.1/2、224.1/2和226.1/2称为:(二元)向上运算符单元222.1/2,也被标记为 Δ ; (二元)向下运算符单元224.1/2,也被标记为 ∇ ;以及一元框运算符单元226.1/2,也被标记为 $\square$ 。为方便起见,已经选取了名称向上、向下和框,但是这些名称自身并不具有意义。运算符222.1/2、224.1/2、226.1/2同样可以被称为第一运算符单元、第二运算符单元和第三运算符单元。向下和向上运算符中的参数可以被指示为下标。

[0140] 二元向上运算符单元222.1/2, Δ 被布置为:

[0141] -接收针对第一域元素(a)和第二域元素(b)的表示,编码的域元素(x)是第一域元素与第二域元素之间的差($x=a-b$),并且接收参数域元素(c),并且

[0142] -计算第一域元素域乘以参数的倒数(ac^{-1})并且计算第二域元素域乘以参数的倒数(bc^{-1}),新编码的域元素(y)由所述两个计算的结果来表示($y=ac^{-1}-bc^{-1}$)。

[0143] 在数学上,所述向上运算符可以由 $\Delta_c((a,b))=(ac^{-1},bc^{-1})$ 来定义。在该公式中,c是参数,并且(a,b)是第一编码的域元素。编码的域元素(a,b)对域元素 $x=a-b$ 进行编码。向上运算符的输出自身也是编码的形式,因此,编码所述元素 $ac^{-1}-bc^{-1}$ 。

[0144] 二元向下运算符单元224.1/2, ∇ 被布置为:

[0145] -接收针对第一域元素(a)和第二域元素(b)的表示,所述编码的域元素(x)是第一域元素与第二域元素之间的差($x=a-b$),并且接收参数域元素(c),并且

[0146] -计算第一域元素域乘以参数(ac)并且计算第二域元素域乘以参数(bc),新编码的域元素(y)由所述两个计算的结果来表示($y=ac-bc$)。

[0147] 在数学上,所述向下运算符可以由 $\nabla_c((a,b))=(ac,bc)$ 来定义,在该公式中,c是参数,并且(a,b)是第一编码的域元素。所述编码的域元素(a,b)对域元素 $x=a-b$ 进行编码。所述向上运算符的输出自身也是编码的形式,因此,编码所述元素 $ac-bc$ 。

[0148] 注意,在技术上,针对向上运算符和向下运算符的参数c以及输入a和b取自第一编码中的域F的范围或者取自第二编码中的 $\mathbb{Z}_{n-1} \cup \{-\infty\}$ 。注意,其具有相同的大小,并且可以在实施例中以相同的方式来表示,例如,相等长度的位串。在实施例中,根据第二实施例的表示中的元素被限制为 $\mathbb{Z}_{n-1}$ 和 $-\infty$ 的单元。乘法集合中的运算符,例如运算符222.2、226.2和224.2,可以适于仅产生由 $\mathbb{Z}_{n-1}$ 和 $-\infty$ 的单元组成的编码。为了保持第一类型的表示的数量相等,可以将第一类型的编码限制为与用于第二类型的相同大小的差异集合,例如,单元数+1。

[0149] 框运算符单元226.1/2,□被布置为:

[0150] -接收针对第一域元素(a)和第二域元素(b)的表示,编码的域元素(x)是第一域元素与第二域元素($x=a-b$)之间的差,

[0151] -以编码的形式获得编码的域元素(x)加上固定增量的取反($-(x+1)=-x-1$)。

[0152] 在数学上,框运算符226.1/2可以由 $\square((a,b))=(k,l)$ 来定义,其中, $k-1=-(x+1)=-x-1$ 。注意,针对(k,l)可能存在多个选项。所述框运算符可以总是选择相同的选项,这可以是随机选项,或者是对编码的表示的特定限制内的选项。下文我们将假设(k,l)两者是单位一或负无穷大。可以使用域加法将框运算符表示为映射(a,b)至(b,a+1)。这种一元运算适合表达为表格。在 $\mathbb{Z}_{n-1}$ 中,其中,一些非零元素可能不是单位,表格能够选择满足任何单位要求的表示(例如,表示的第一元素是单位一,第二元素是单位一,或者这两个元素是单位一)。此外,表格不需要遵循任何特定的公式表示为针对所有输入的基本表达式(仅涉及域加法、减法、乘法和乘法逆);并且具体地,不需要遵循上文给出的表达。

[0153] 运算符222.1/2、224.1/2和226.1/2中的所有运算符可以被实施为表格。针对框运算符,这是自然的选择。向上和向下运算符222.1/2和224.1/2也可以使用域算法来实施,例如,由计算设备混淆的相同域算法。

[0154] 每当运算符222.2、224.2的输入或参数中的任何一个负无穷大时,结果可以被定义为例如点 $(-\infty, -\infty)$ 。每当框运算符226.2的输入中的任何一个输入是负无穷大时,所述结果可以被定义为例如点 $(-\infty, -\infty)$ 、 $(-\infty, a)$ 或 $(a, -\infty)$,针对特定整数 $a \neq -\infty$ 。

[0155] 通常,每当乘法集合129、229的运算符的输入/参数是中的任何一个输入/参数是负无穷大时,通过将至少一个输出元素设置为负无穷大,这确保了每当由计算管理器130乘以的第一编码的域元素和第二编码的域元素中的任何一个表示0并且具有负无穷大元素时,应用乘法集合的最终结果也将具有负无穷大元素并且因此表示0。简言之,总是在输入中向输出传播负无穷大确保乘以0将得到0作为结果。

[0156] 给定分别被表示为(a,b)和(c,d)的第一编码的域元素和第二编码的域元素(均针对相同类型来编码),下文给出编码中域加法的序列。效果将是:

[0157] -如果根据第一编码对第一编码的域元素和第二编码的域元素进行编码,则结果

是根据第一编码而编码的加法，

[0158] -如果根据第二编码对第一编码的域元素和第二编码的域元素进行编码，则结果是根据第二编码而编码的乘法(从第二编码的角度看，相加了两个表示)。

[0159] 第一编码的域元素和第二编码的域元素的元素可以被称为第一第一域元素(a)、第一第二域元素(b)、第二第一域元素(c)和第二第二域元素(d)。

[0160] 所述域加法可以被表示如下：

$$[0161] \quad (a, b) + (c, d) = (\nabla_d \circ \square \circ \Delta_d \circ \nabla_c \circ \square \circ \Delta_c)((a, b))$$

[0162] 在该公式中，函数组合利用 $\circ$ 来指示。例如，

[0163] -首先，使用参数c将向上运算符单元222应用于第一编码的域元素(a,b)，然后

[0164] -将框运算符226应用于前一运算符的结果，然后

[0165] -使用参数c将向下运算符单元224应用于前一运算符的结果，然后

[0166] -使用参数d将向上运算符单元222应用于前一运算符的结果，然后

[0167] -将框运算符226应用于前一操作符的结果，然后

[0168] -使用参数d将向下运算符单元224应用于前一运算符的结果。

[0169] 运算符引用的序列可以是例如224、226、224、222、226、222(在这种情况下，第一运算符在左侧)。参数的序列可以是例如c、-、c、d、-、d，其中，没有参数则利用连字符来指示。

[0170] 注意，针对根据第一编码而编码的域元素，可以使用相同的运算符来定义乘法。域乘法可以被表示如下：

$$[0171] \quad (a, b) \cdot (c, d) = (\nabla_{ad} \circ \square \circ \Delta_{ad} \circ \nabla_{bd} \circ \square \circ \Delta_{cbd})((a, b))$$

[0172] 例如，

[0173] -首先，使用参数cbd将向上运算符单元222应用于第一编码的域元素(a,b)，然后

[0174] -将框运算符226应用于前一运算符的结果，然后

[0175] -使用参数bd将向下运算符单元224应用于前一运算符的结果，然后

[0176] -使用参数ad将向上运算符单元222应用于前一运算符的结果，然后

[0177] -将框运算符226应用于前一运算符的结果，然后

[0178] -使用参数ad将向下运算符单元224应用于前一运算符的结果。

[0179] 运算符引用的序列可以是例如224、226、224、222、226、222(在这种情况下，第一运算符在左侧)。参数的序列可以是例如cbd、-、bd、ad、-、ad，其中，没有参数则利用连字符来指示。

[0180] 尽管当关于表格使用方面该操作也是非常相似的，但是可以根据需要在域级上进行一些乘法的事实来收集一些信息。转换为第二编码避免了该问题。

[0181] 可以例如通过交换第一表示中的元素，来通过对所述元素中的一个元素取反来实现减法。可以通过乘以倒数来实现除法。可以通过交换第二表示中的元素来获得倒数。注意，元素0以这种方式被映射到0。

[0182] 备选地，可以通过分别在第一编码中执行来获得域减法：

$$(a, b) - (c, d) = (\nabla_c \circ \square \circ \Delta_c \circ \nabla_d \circ \square \circ \Delta_d)((a, b)) \circ$$

[0183] 例如，

[0184] -首先，使用参数d将向上运算符单元222应用于第一编码的域元素(a,b)，然后

[0185] -将框运算符226应用于前一运算符的结果，然后

[0186] -使用参数d将向下运算符单元224应用于前一运算符的结果,然后

[0187] -使用参数c将向上运算符单元222应用于前一运算符的结果,然后

[0188] -将框运算符226应用于前一运算符的结果,然后

[0189] -使用参数c将向下运算符单元224应用于前一运算符的结果。

[0190] 运算符引用的序列可以是例如224、226、224、222、226、222 (在这种情况下,第一运算符在左侧)。参数的序列可以是例如d、-、d、c、-、c,其中,没有参数则利用连字符来指示。

[0191] 以相同的方式,能够使用第二表示来获得除法。

[0192] 可以设想到具有能够分别表示在第一编码和第二编码中的加法和乘法的属性的其他序列。例如,通过在上述公式中替换作为框运算符和向上运算符的组合($\square \circ \Delta$)的新运算符单元,来获得新的序列。通过将 $\nabla_c \circ \square \circ \Delta_c$ 组合到单个二元运算符而获得了有趣的变型。在这种情况下,不需要框操作。在这种情况下,加法和乘法集合仅具有一个元素。存在许多其他可能性。在实施例,集合128和129中的每个包括多个元素。

[0193] 可以通过替换运算符单元的对应定义并且遵循数学推导来验证以上公式是正确的。

[0194] 针对所述编码存在不同的选项。例如,可以将编码中的两个域元素的列表表示为指数 $((\alpha, \beta))$,两个域元素是所述域的公共基本元素(u)的指数,其被提升到由指数 $(u^\alpha - u^\beta)$ 指示的幂。该表示被称为对数形式,可以添加小写u以区分该表示。选择基本元素作为元素,使得u的幂的集合是针对F(在第一类型的编码的情况下)或者在第二类编码的情况下是 $\mathbb{Z}_{n-1} \cup \{-\infty\}$ 的差异集合。我们使用 $u^{-\infty} = 0$ 的约定。

[0195] 发生器将是针对基本元素u的可能选择,然而,其是不需要的。例如,考虑在上文的范例中引入256的域,形式变量x不是GF(256)的发生器,因为 $x^{55} = 1$,然而,x的幂清楚地形成差异集合。

[0196] 针对域元素的不同编码可以给出针对运算符的不同的公式。例如,如果编码是以对数形式,则针对向上、向下和框运算符的以上描述可以适于类似的公式。另一变型是将元素表示为x,其具有对数形式 $(\alpha, \beta)_u$ 作为域元素的列表 $[\beta, \delta]$,其中, $\delta = \alpha - \beta$,亦即, $x = u^{\beta + \delta} - u^\beta$ 。同样地,在这种形式中,可以表达运算符。例如,所述向上运算符将变为 $\Delta_\gamma([\beta, \delta]) = [\beta - \gamma, \delta]$ 。所述向下运算符将是相同的,除了以 $+\gamma$ 替代 $-\gamma$ 。可以使用与之前相同的关系来定义框操作。

[0197] 存在运算符的其他序列,其也可以被用于创建运算符的序列,使得针对域乘法的序列与针对域加法的序列相同。例如,在实施例,定义2个、3个或更多个运算符对F的元素的列表表示的元素进行操作。所述运算符中的一些运算符(例如,1个或更多个,或者2个或更多个,或者除了一个之外的所有)是在F的两个元素的列表表示的元素上操作的域操作的预先定义的序列。域操作的所述预先定义的序列可以被表示为涉及域乘法和域乘法逆运算的表达式,并且任选地还涉及域加法和域减法。运算符中的一些运算符(例如,恰好是1个,1个或更多个等)是一元的并且表示对所述域元素的固定运算,例如,涉及域加法和加法逆(减号,“-”,运算)的固定表达;该运算可以被表示为表格运算。尽管没有必要。在编码的实施例,可以对列表表示的元素进行编码。然后,可以将域运算、加法、乘法、乘法和加法逆等实施为编码的表格或表格网络。所述列表表示可以被定义为域F的元素的序列,以及从列表表示到F的满射映射。例如,映射到 $(a-b)$ 的差异表示 (a, b) 是一种这样的列表表示,在本

文中给出了其他范例。

[0198] 下文给出了运算符的另外的序列,其也可以被用于创建运算符的序列,从而针对域乘法的序列与针对两个编码中的域加法的序列相同。

[0199] 例如,令 F 乘以域,并且令 $a, b, c, d \in F$ 。使用解释 $x = a - b; y = c - d$ 将两个元素 x 和 y 表示为对 (a, b) 和 (c, d) 。注意以下推导:

$$[0200] \quad x + y = (a - b) + (c - d)$$

$$[0201] \quad = a + c - b - d$$

$$[0202] \quad = (ab^{-1} + cb^{-1} - 1) b - d$$

$$[0203] \quad = (e - f) b - d$$

$$[0204] \quad = eb - fb - d$$

$$[0205] \quad = -1 (fb + d - eb)$$

$$[0206] \quad = -1 (fbb^{-1}e^{-1} + db^{-1}e - 1) eb$$

$$[0207] \quad = -1 (g - h) eb$$

$$[0208] \quad = (h - g) eb$$

$$[0209] \quad = (heb - geb)$$

[0210] 后者可以再次被写为对 (heb, geb) 。在该范例中,我们定义了框运算符

$$\square(a, b) = a + b - 1 = e - f \circ \text{针对计算 } (a, b) + (c, d), \text{ 首先,我们计算 } (x_1, x_2) = \nabla_b \square \Delta_b((a, c)) \circ$$

由此, $(a, b) + (c, d)$ 的结果可以被计算为 $-\nabla_{x_1} \square \Delta_{x_1}((x_2, d)) \circ$ 可以使用相同的运算符但是在第二编码中执行乘法。该表示也可以适用于在第一编码类型中的乘法,使用标识 $(a, b) \cdot (c, d) = (ac, ad) + (bd, bc)$ 。

[0211] 在实施例中,可以使用不同的编码对域 F 的元素进行编码。在以下范例中,混合了域元素的两个不同表示。令 $x = [\alpha, \Delta_0]_u = u^{\alpha + \Delta_0} - u^\alpha$, 并且 $y = (\sigma, \tau)_u = u^\sigma - u^\tau$ 。我们将框运算符(在本范例中标记为 $\boxplus$)定义为 $\boxplus(\alpha, \Delta_0) = [\beta, \Delta_1]_u$, 其中, $u^{\Delta_0} + u^\alpha - 1 = u^\beta - u^{\beta + \Delta_1}$ 。在以下表格中,推导被显示在左侧,并且运算符显示在右侧。

	$x + y = (u^{\alpha + \Delta_0} - u^\alpha) + (u^\sigma - u^\tau)$	$[\alpha, \Delta_0]_u + (\sigma, \tau)_u$
	$= u^{\alpha + \Delta_0} - u^\alpha + u^\sigma - u^\tau$	
	$= u^\alpha (u^{\Delta_0} + u^{\sigma - \alpha} - 1) - u^\tau$	$\Delta_\alpha(\sigma, \tau)_u = (\sigma - \alpha, \tau - \alpha)_u$
	$= u^\alpha (u^\beta - u^{\beta + \Delta_1}) - u^\tau$	$\boxplus(\sigma - \alpha, \Delta_0) = (\beta, \Delta_1)_u$
	$= u^{\alpha + \beta} - u^{\alpha + \beta + \Delta_1} - u^\tau$	
[0212]	$= -1(u^{\alpha + \beta + \Delta_1} + u^\tau - u^{\alpha + \beta})$	$\Delta_\beta \circ \Delta_\alpha(\sigma, \tau)_u = (\sigma - \alpha - \beta, \tau - \alpha - \beta)_u$
	$= -1 \cdot u^{\alpha + \beta} (u^{\Delta_1} + u^{\tau - \alpha - \beta} - 1)$	$\boxplus(\tau - \alpha - \beta, \Delta_1) = (\gamma, \Delta_2)_u$
	$= -1 \cdot u^{\alpha + \beta} (u^\gamma - u^{\gamma + \Delta_2})$	
	$= -1(u^{\alpha + \beta + \gamma} - u^{\alpha + \beta + \gamma + \Delta_2})$	
	$= u^{\alpha + \beta + \gamma + \Delta_2} - u^{\alpha + \beta + \gamma}$	
	$= (\alpha + \beta + \gamma, \Delta_2)_u$	$\nabla_\alpha \circ \nabla_\beta \circ \boxplus(\tau - \alpha - \beta, \Delta_1)$

[0213] 计算设备100可以包括输入接口,所述输入接口被布置为接收第一编码的域元素和第二编码的域元素,所述第一编码的域元素和所述第二编码的域元素两者是根据第一编码或第二编码来编码的。所述输入接口可以采用各种形式,诸如到局域网或广域网(例如,互联网)的网络接口,到内部数据存储装置或外部数据存储装置的存储接口等。

[0214] 通常,设备100包括微处理器(在图1中未单独地示出),其执行被存储在设备100中的适当软件;例如,该软件可能已经被下载和/或存储在对应的存储器中,例如,诸如RAM的易失性存储器或者诸如闪存的非易失性存储器(未单独地示出)。备选地,设备100可以被全部地或部分地以可编程逻辑实施为例如现场可编程门阵列(FPGA)。设备100可以被全部地或部分地实施为所谓的专用集成电路(ASIC),即,被实施为针对其特定用途定制的集成电路(IC)。例如,所述电路可以以CMOS来实施,例如,使用诸如Verilog、VHDL等的硬件描述语言。

[0215] 在实施例中,设备100包括操作数存储电路、运算符模块电路和计算管理器电路。所述电路实施在本文中所描述的对应单元。所述电路可以是处理器电路和存储电路,所述处理器电路执行在所述存储电路中电子地表示的指令。所述电路也可以是FPGA、ASIC等。操作数存储电路、运算符模块电路可以是电子存储装置,例如电子存储器。

[0216] 图3图示了被布置为对域的元素执行计算的计算方法300作为示意性流程图。所述方法包括:

[0217] -存储310编码的域元素(112、114、116;212),编码的域元素以编码的形式来表示所述域的元素,其中,编码的域元素是根据至少两种不同的编码中的一种编码来编码的。例如,所述编码的域元素可以被存储在存储器中,例如,易失性存储器,例如云存储等。

[0218] -至少两种不同编码的第一编码,其中,所述编码的域元素(x)被表示,

[0219] -至少两种不同编码的第二编码,其中,所述编码的域元素被表示为指数(s),所述编码的域元素是域的发生器(g)的指数的幂($x = g^s$),

[0220] 所述方法还包括:

[0221] -接收320第一编码的域元素和第二编码的域元素,所述第一编码的域元素和所述第二编码的域元素两者是根据第一编码或第二编码来编码的,这些编码元素可以是在存储310期间存储的元素中的一些元素。

[0222] 所述方法选择330是相加还是相乘第一编码的域元素((a,b))和第二编码的域元素((c,d))。例如,可以执行运算符的序列,例如,作为计算机程序的部分。所述方法还包括:

[0223] -对于相加:将第二转换运算符单元应用342到根据第二编码而编码的任何编码的域元素,并且应用344加法运算符单元的集合,所述第二转换运算符单元被布置为接收根据第二编码而编码的域元素,并且被布置为产生根据第一编码而重新编码的相同域元素,加法运算符单元的集合(128)被布置为将根据第一编码而编码的两个域元素相加,

[0224] -对于相乘:将第一转换运算符单元应用352到根据第一编码而编码的任何编码的域元素,并且应用354乘法运算符单元的集合,所述第一转换运算符单元被布置为接收根据第一编码而编码的域元素,并且被布置为产生根据第二编码而重新编码的相同域元素,乘法运算符单元的集合(129)被布置为通过将指数相加来将根据第二编码而编码的两个域元素相乘。

[0225] 对于本领域技术人员将显而易见的是,执行所述方法的许多不同方式都是可能

的。例如,能够改变步骤的次序,或者可以并行执行一些步骤。此外,在步骤之间可以插入其他方法步骤。所插入的步骤可以表示诸如在本文中所描述的方法的改进,或者可以与所述方法无关。在下一步骤开始之前,给定步骤可能尚未完全地完成。

[0226] 可以使用软件来执行根据本发明的方法,所述软件包括用于使处理器系统执行方法300的指令。软件可以仅包括由所述系统的特定子实体采取的那些步骤。所述软件可以被存储在适合的存储介质中,诸如硬盘、软盘、存储器、光盘等。所述软件可以作为信号沿着电线、或者无线地或者使用数据网络(例如,互联网)来发送。所述软件可供下载和/或在服务器上远程地使用。可以使用比特流来执行根据本发明的方法,所述比特流被布置成配置可编程逻辑(例如,现场可编程门阵列(FPGA))以执行所述方法。

[0227] 将意识到,本发明还扩展到计算机程序,特别是适用于将本发明付诸实践的载体上或载体中的计算机程序。所述程序可以是源代码、目标代码、代码中间源以及诸如部分编译形式的目标代码的形式,或者是适用于实施根据本发明的方法的任何其他形式。涉及计算机程序产品的实施例包括与所述方法中的至少一种方法的处理步骤中的每个处理步骤相对应的计算机可执行指令。这些指令可以被细分为子例程和/或被存储在可以静态地或动态地链接的一个或多个文件中。涉及计算机程序产品的另一实施例包括与所述系统和/或产品中的至少一个的每个单元相对应的计算机可执行指令。

[0228] 图4a示出了根据实施例的计算机可读介质1000,其具有包括计算机程序1020的可写部分1010,所述计算机程序1020包括用于使处理器系统执行计算方法的指令。计算机程序1020可以作为物理标记或者借助于对计算机可读介质1000的磁化在计算机可读介质1000上实现。然而,也可以设想到任何其他合适的实施例。此外,将意识到,尽管计算机可读介质1000在此被示为光盘,但是计算机可读介质1000可以是任何适合的计算机可读介质,诸如硬盘、固态存储器、闪存等,并且可能不可录制或可录制。计算机程序1020包括用于使处理器系统执行所述计算方法的指令。

[0229] 图4b示出了根据实施例的处理器系统1140的示意性表示。所述处理器系统包括一个或多个集成电路1110。一个或多个集成电路1110的架构在图4b中示意性地示出。电路1110包括处理单元1120,例如,CPU,用于运行计算机程序部件以执行根据实施例的方法和/或实施其模块或单元。电路1110包括用于存储编程代码、数据等的存储器1122。存储器1122的部分可以是只读的。电路1110可以包括通信元件1126,例如,天线、连接器或者这两者等。电路1110可以包括专用集成电路1124,其用于执行在所述方法中定义的处理的部分或全部处理。处理器1120、存储器1122、专用IC1124和通信元件1126可以经由互连1130(例如,总线)被连接到彼此。处理器系统1110可以被布置用于分别使用天线和/或连接器进行接触式和/或非接触式通信。

[0230] 例如,在实施例中,所述计算设备可以包括处理器电路和存储器电路,所述处理器其被布置为执行被存储在存储器电路中的软件。例如,所述处理器电路可以是Intel Core i7处理器、ARM Cortex-R8等。所述存储器电路可以是ROM电路,或者是非易失性存储器,例如,闪存。所述存储器电路可以是易失性存储器,例如,SRAM存储器。所述计算设备可以包括非易失性软件接口,例如,硬盘驱动器、网络接口等,其被布置用于提供软件。所述处理器电路可以包括协作执行所述软件的多个处理器核心。

[0231] 以下条款涉及本发明的各种实施例。申请人由此通知,在本申请或由此衍生的任

何另外的申请的审查期间,可以对这样的条款和/或从说明书中获得的这样的条款和/或特征的组合制定新的权利要求。

[0232] 条款1:一种计算设备(100),其被布置为对域(F)的元素执行计算,域加法和域乘法是在所述域上定义的,所述计算设备包括:

[0233] -操作数存储(110),其被布置为存储编码的域元素(112、114、116;212),编码的域元素表示编码的形式的所述域的元素,其中,编码的域元素是根据至少两种不同的编码中的一种编码来编码的;

[0234] -所述至少两种不同编码的第一编码,其中,所述编码的域元素(x)被表示,

[0235] -所述至少两种不同编码的第二编码,其中,所述编码的域元素被表示为指数(s),所述编码的域元素是域的发生器(g)的指数的幂($x=g^s$),-包括多个运算符单元的运算符模块(120;220),所述多个运算符单元包括:

[0236] -第一转换运算符单元,其被布置为接收根据所述第一编码而编码的域元素,并且被布置为产生根据所述第二编码而重新编码的相同域元素,

[0237] -第二转换运算符单元,其被布置为接收根据所述第二编码而编码的域元素,并且被布置为产生根据所述第一编码而重新编码的相同域元素,

[0238] -加法运算符单元的集合(128),所述加法运算符单元的集合被布置为将根据所述第一编码而编码的两个域元素相加,以及

[0239] -乘法运算符单元的集合(129),所述乘法运算符单元的集合被布置为通过将所述指数相加来将根据所述第二编码而编码的所述两个域元素相乘,

[0240] -计算管理器(130),其被布置为:

[0241] -接收第一编码的域元素和第二编码的域元素,所述第一编码的域元素和所述第二编码的域元素两者是根据第一编码或第二编码来编码的,

[0242] -通过以下操作选择性地将第一编码的域元素((a,b))和第二编码的域元素((c,d))相加或相乘:

[0243] -对于相加:将所述第二转换运算符单元应用于根据所述第二编码而编码的任何编码的域元素,并且应用加法运算符单元的集合,

[0244] 并且

[0245] -对于相乘:将所述第一转换运算符单元应用于根据所述第一编码而编码的任何编码的域元素,并且应用所述乘法运算符单元的集合。

[0246] 条款2:11、根据条款1的计算设备,其中,在第一编码和/或第二编码中的两个域元素的列表被表示为指数((α,β)),所述两个域元素是所述域的公共基本元素(u)的指数增加到由指数指示的幂($x=u^\alpha-u^\beta$)。

[0247] 应当注意,上文提到的实施例说明而非限制本发明,并且本领域技术人员将能够设计许多备选实施例。

[0248] 在权利要求中,被放置在括号之间的任何参考符号不应当被解释为限制权利要求。动词“包括”以及其变形的使用不排除存在权利要求中所述之外的元件或步骤。在元件前面的冠词“一”或“一个”不排除存在多个这样的元件。本发明可以借助于包括若干不同元件的硬件以及借助于适当编程的计算机来实施。在列举了若干单元的设备权利要求中,这些单元中的若干个单元可以由同一个硬件项来体现。在相互不同的从属权利要求中叙述特

定措施的仅有事实并不指示不能有利地使用这些措施的组合。

[0249] 在权利要求中,在括号中的参考文献参考实施例的附图中的附图标记或实施例的公式,从而增加了权利要求的可理解性。这些参考文献不应当被解释为限制权利要求。

100

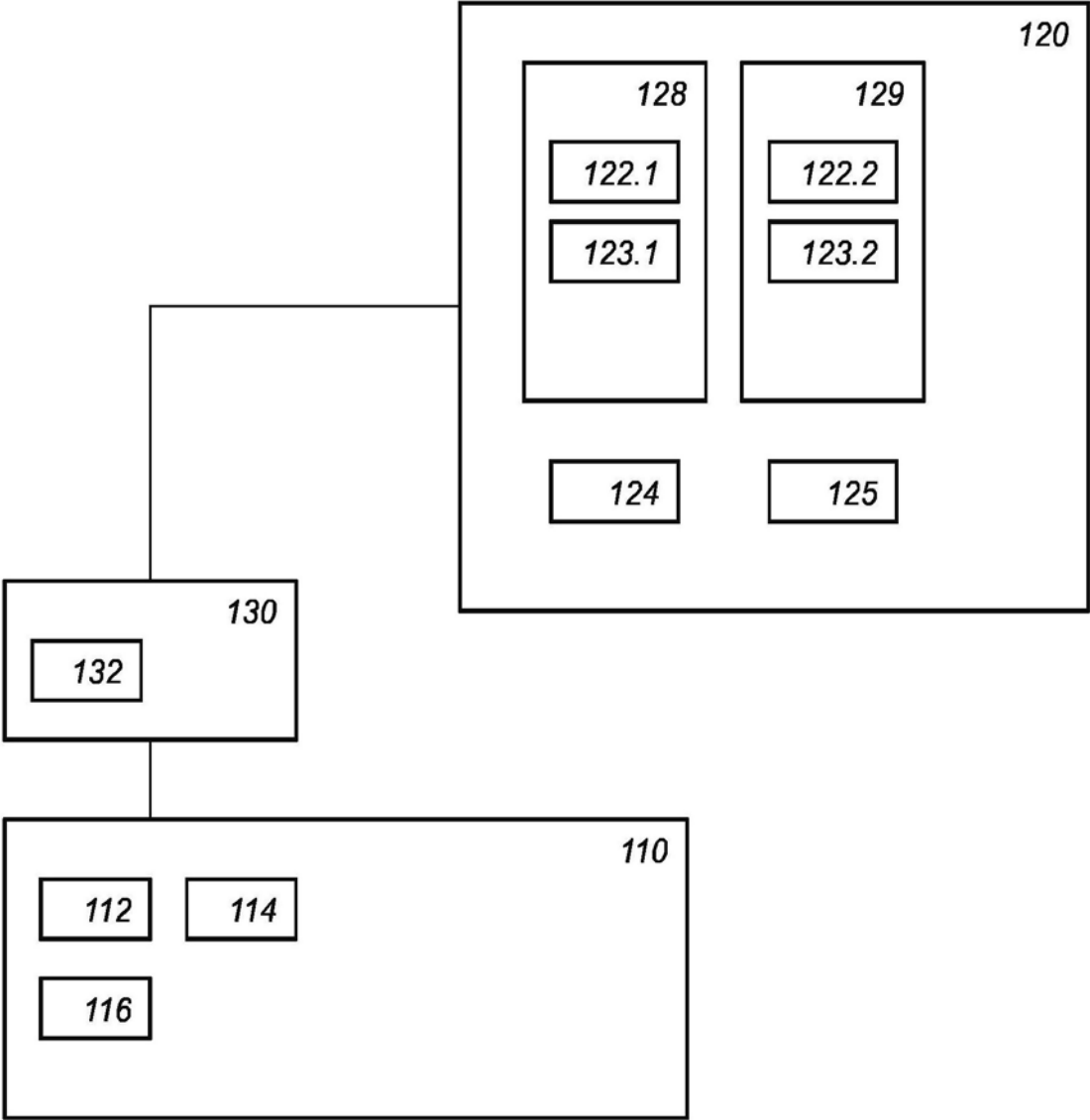


图1

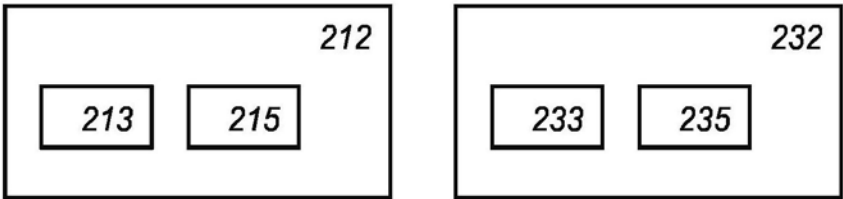


图2a

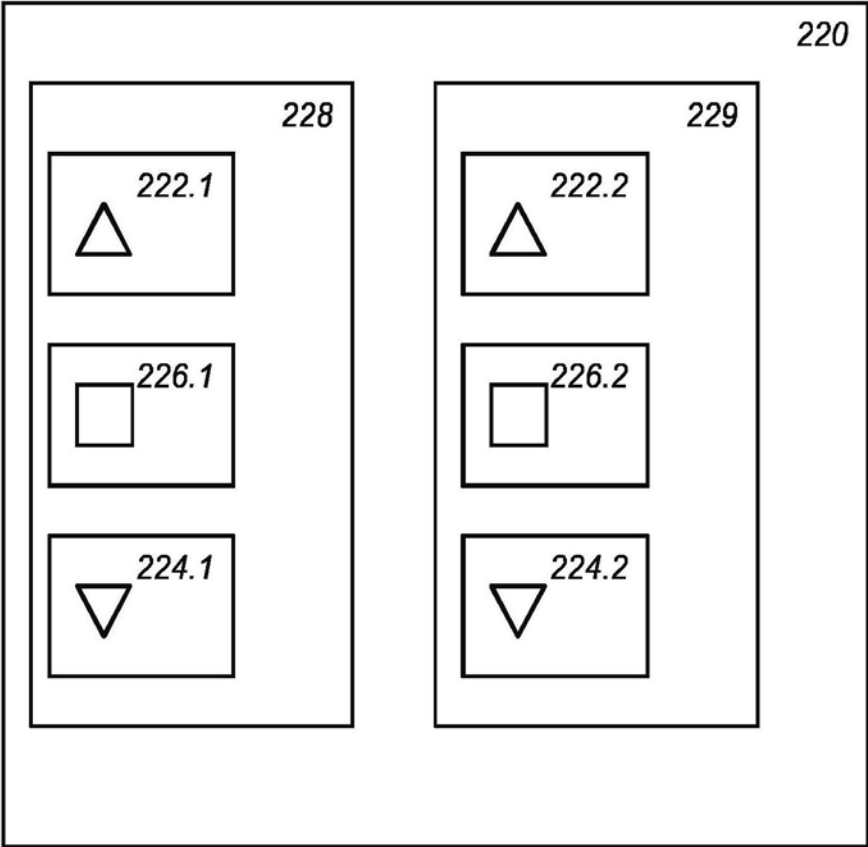


图2b

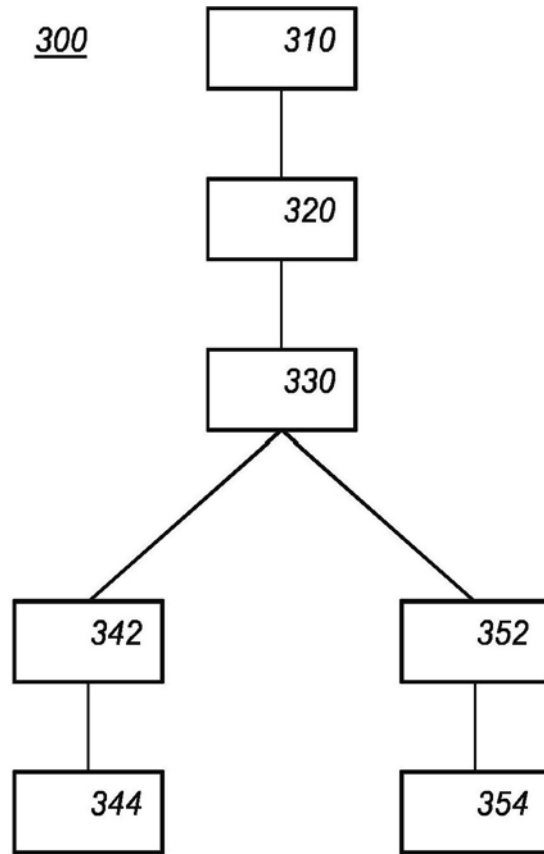


图3

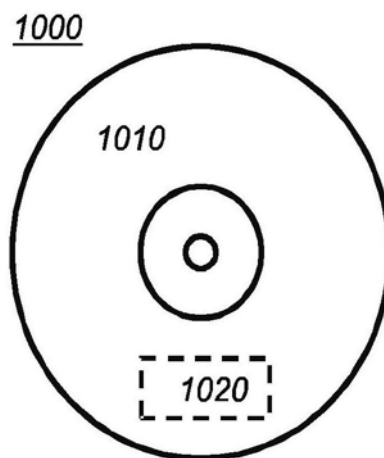


图4a

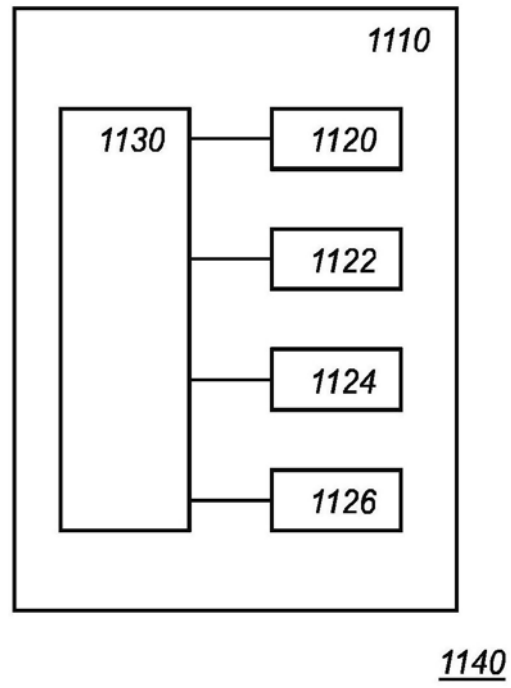


图4b