文章编号:1001-506X(2010)05-1094-06

基于自组装 DNA 计算的 RSA 密码系统破译方案

张勋才[1,2], 牛 莹[1], 崔光照[1], 许 进[2]
(1. 郑州轻工业学院电气信息工程学院, 河南 郑州 450002;
2. 华中科技大学控制科学与工程系, 湖北 武汉 430074)

摘 要: 自组装 DNA 计算在解决 NP 问题,尤其是破译密码系统方面,具有传统计算机无法比拟的优势。采用 DNA 分子瓦编码信息,借助于分子瓦之间的粘性末端进行自组装,给出了乘法运算的实现方案。在此基础上,通过引入非确定性的指派分子瓦,提出了一种用自组装 DNA 计算破译 RSA 公钥密码系统的非确定性算法。通过创建数以亿计的参与计算的 DNA 分子瓦,在 DNA 计算能力允许的范围内,该算法可以并行地测试每个可能的因子,以高概率地分解整数。该方法最大的优点是充分利用了 DNA 分子瓦具有的海量存储能力、生化反应的巨大并行性以及组装的自发有序性。

关键词: 自组装; DNA 分子瓦; 非确定性计算; 整数分解; RSA
中图分类号: TP 309.7 **文献标志码**: A **DOI**:10.3969/j.issn.1001-506X.2010.05.045

Breaking the RSA public key cryptosystem using self-assembly of DNA tilings

ZHANG Xun-cai[1,2], NIU Ying[1], CUI Guang-zhao[1], XU Jin[2]
(*1. Coll. of Electrical and Electronic Engineering, Zhengzhou Univ. of Light Industry, Zhengzhou 450002, China;*
2. Dept. of Control Science and Engineering, Huazhong Univ. of Science and Technology, Wuhan 430074, China)

Abstract: Computation by self-assembly of DNA is an efficient method of executing parallel DNA computing where information is encoded in DNA tiles and a large number of tiles can be self-assembled via sticky end associations. This paper shows that how the DNA self-assembly process can be used for breaking the RSA public key cryptosystem, whose security is based on the difficulty of factoring the product of two large prime numbers. Thus, a method for implementing the product of two primers using self-assembled DNA computing is expounded. Then, a non-deterministic algorithmic is provided to break efficiently the RSA public key cryptosystem. By creating billions of copies of the participating DNA tiles, the algorithmic will run in parallel on all possible factors. The computation takes advantage of non-determinism, but theoretically, each of the non-deterministic paths is executed in parallel, yielding the solution in time linear in the size of the input, with high probability. It presents clear evidence of the ability of molecular computing to perform complicated mathematical operations.

Keywords: self-assembly; DNA tile; non-deterministic computation; integer factorization; Rivest-Shamir-Adleman (RSA)

0 引 言

1994 年,Adleman 博士首次在实验室借助 DNA 分子,通过可控的生化反应成功解决了一个具有七顶点的有向 Hamilton 路问题[1],这标志着 DNA 计算模式的诞生。同时向人类揭示,DNA 分子可作为计算的介质用于解决的数学问题。这一全新的计算模式,在计算科学领域中产生了极其深远的影响。

DNA 计算是通过控制 DNA 分子间的生化反应来完成运算,最大优点是充分利用了 DNA 分子所具有的海量存储能力以及生化反应的巨大并行性。作为一种新的计算模式,DNA 计算引起众多专家和学者的关注,迅速成为活跃的研究领域。特别是在近几年,利用 DNA 计算来解决各种 NP 完全问题[2-3]、破译和分析现行的密码体制以及研制通用 DNA 计算机等都是 DNA 计算研究的范畴[4-5]。但一个比较明显的困难是这些计算方法所需要的实验操作次数每增加

收稿日期:2008-12-02; **修回日期**:2009-06-12。
基金项目:国家自然科学基金(60773122,60803113);国家高技术研究发展计划(863 计划)(2006AA01Z104);郑州轻工业学院博士科研基金(2009BSJJ006)资助课题
作者简介:张勋才(1981-),男,博士研究生,主要研究方向为 DNA 计算与信息安全、系统工程。E-mail:zhangxuncai@163.com

一次都会带来更多的时间消耗和误差倾向;并且随着问题规模的增大,需要的操作次数也越来越多。1995 年,Winfree 提出了利用 DNA 分子瓦自组装过程进行计算的重要思想[6],进一步为 DNA 计算领域的发展奠定了坚实的理论与实验基础。自组装 DNA 计算的思想组合了 DNA 计算[1]、Tiling 理论[7]和纳米技术[8]。自组装 DNA 计算的一个明显优势就是它可以同时创造数以百万计的同一种结构的拷贝,这和其分子处理过程的固有并行性有关。从概念上来说,算法自组装能用最简单的形式处理任意复杂的信息。

数据的加密与解密是计算机特别是超级计算机应用的一个重要领域,而这一领域正被 DNA 计算机研究者们所关注[4,9]。RSA 密码是目前最先进、最高效的公钥密码之一,其安全性是基于大整数素因子分解的困难性之上。本文针对 RSA 密码系统的特点,给出用自组装 DNA 计算实现整数的乘法运算,并提出一种破译 RSA 密码系统的非确定性算法。

1　DNA 分子自组装与 RSA 密码系统

1.1　DNA 分子自组装

分子自组装的理论研究始于 1961 年 Wang 提出的多米诺分子瓦拼接的概念[10],它表明具有一定规则和不同颜色边的多边形集合,根据相同颜色的边可以组装到一起的加法规则,来组装成一个平面或者网格,进而完成整个计算过程,则这个集合就是图灵等价的。我们可以在 DNA 分子的粘贴末端分支和 Wang Tiles 理论中多边形的不同着色的边建立一一对应的关系。1980 年以后,Seeman 一直致力于 DNA 分子瓦构造方面的研究,对分子瓦结构的物理、化学性质作了深入细致的探索,并构造出了多种 DNA 分子纳米结构[11-12]。当前用于实现自组装 DNA 计算的分子瓦包括 DX(double-crossove)分子[13-14],TX(triple-crossover)分子[15-16]以及 HJ(holliday junction analogues)结构[17]。本文中,我们将每个 DNA 分子瓦抽象为一个带有标签的多边形,每个标签标识一个特别的粘性末端。具有互补特性的两个粘性末端可以相互连接到一起。每个分子瓦可以有 1～6 个粘性末端(见图 1)。这些分子瓦能用各种不同的 DNA 编码序列进行构建(可参考与编码序列设计相关的文献)。并且,不同的序列能标识不同的符号。

图 1　抽象的 DNA 分子瓦示意图

1994 年,Adleman 用自组装 DNA 计算的一种简单形式,给出了第一个采用 DNA 分子的自组装计算实验[1]。1995 年,Winfree 在 Seeman 的分子纳米结构理论基础上,提出了通过分子瓦构造纳米结构的过程来实现计算的思想,并证明了其计算的完备性[13],随后逐渐发展成以分子瓦自组装为基础的 DNA 计算。1998 年,Winfree 提出了 DNA 分子瓦自组装的动力学模型。2000 年,Adleman 建立了分子瓦自组装的随机微分方程模型,并确定了一维自组装的均衡概率分布收敛速度[18]。2002 年,Adleman 给出了树状自组装的程序复杂度的界限[3]。2004 年,Rothemund 等报道了利用 DX 分子瓦实现一维元胞自动机的一个实验结果,并证明用 DNA 分子瓦自组装实现任何元胞自动机的可能性[19]。Brun 从理论上给出了基于 DNA 分子瓦自组装的二进制加法和乘法运算系统[20],并在此基础上解决整数分解与子集和问题[21-22]。我们给出了实现二进制减法和除法的运算系统[23]。

DNA 分子自组装是指在一定的温度、浓度、酸碱度以及特定酶的作用下,一些带有输入信息的 DNA 分子根据 Watson-Crick 互补配对原则,自组装成新的带有输出信息的 DNA 分子的过程。DNA 分子瓦自组装是可编程的。如前所述,我们仅用如下四步即可完成一种计算过程:(1) 混合、输入寡核苷酸;(2) 使分子瓦经自组装生成超级结构;(3) 链接所加入的分子瓦;(4) 分离(提纯)以检验结果[24]。

1.2　RSA 密码系统及其安全性

当前最著名、应用最广泛的 RSA 密码系统在 1978 年由美国麻省理工学院的 Rivest、Shamir 和 Adleman 提出。对 RSA 密码系统的攻击可能有如下三种方式:分解整数 N、确定欧拉函数 $\Phi(N)$ 和直接获得私钥 d。可以证明,后两种攻击方式均等价于第一种方式。因此,目前对 RSA 密码的分析大都集中于第一种攻击方法,即将整数 N 分解为两个素因子。

由于 DNA 分子具有独特的数据机构和并行操作的能力,建立在分子水平上的 DNA 计算机具有极其诱人的研究前景,尤其在破译密码系统方面。对于 RSA 密码系统,攻击者可以通过分解整数 N 来破译。Chang 等人利用 DNA 计算给出了一种整数分解的方法[25];Beaver[26] 等人借助于 Adleman 的思想将整数分解问题转化为 Hamiltonian 路问题来解决。然而这两个算法生物操作步数随着整数位数的增加而迅速增多。为此,根据自组装 DNA 计算的思想[6],Brun 于 2008 年从理论上提出了一种整数分解模型[21]。下面我们给出另一种基于自组装 DNA 计算的 RSA 密码系统破译方案。

2　破译 RSA 密码系统

2.1　自组装 DNA 计算实现乘法运算

乘法运算是一种很重要的运算,传统计算机采用硬件乘法器通过移位操作和加法操作直接实现。记一个 k 位无符号二进制数 P 为 $p_{k-1}p_{k-2}\cdots p_1p_0$,其中 p_i 仅取 0 或 1。常规乘法运算模式如表 1 所示,它描述了两个四位二进制数相乘的情形。设整数 N 为给定的两个 k 位二进制整数 P(被乘数)和 Q(乘数)的积,那么从表 1 我们可以看出整数 N 的前 i 位由整数 P 和 Q 的前 $i(0\leqslant i\leqslant k)$ 位决定,这为我们后面所述的非确定性思想提供了思路。自组装后 DNA 分子的空间排列和执行乘法的电路是非常相似的[27]。Brun 利用自组装模型给出了一个乘法的实现。该方法仅适用于理论上的研究。Pelletier 给出了另一种自组装模型

实现乘法运算的方案，并详细分析了系统复杂度[28]；但是由于该方案中的分子瓦还没能有效设计，距离生物实现比较遥远。这里我们借助于 Pelletier 的思想，提供一种可生物实现的乘法运算方案。

表 1 常规乘法运算模式

	n_0	n_1	n_2	n_3	n_4	n_5	n_6	n_8
								+
p_3	0	0	0	p_3*q_0	p_3*q_1	p_3*q_2	p_3*q_3	
p_2	0	0	p_2*q_0	p_2*q_1	p_2*q_2	p_2*q_3	0	
p_1	0	p_1*q_0	p_1*q_1	p_1*q_2	p_1*q_3	0	0	
p_0	p_0*q_0	p_0*q_1	p_0*q_2	p_0*q_3	0	0	0	
×	q_0	q_1	q_2	q_3				

假定我们用 DNA 分子瓦来描述给定的二进制数，每个分子瓦表示该二进制数的一位。通过将输入数表示成分子瓦串，二进制乘法的计算规则将能用一系列的规则分子瓦来实现。图 2 给出了相应的 DNA 分子瓦结构示意图。图 2(a)所示的分子瓦结构用于描述 P 的二进制位，该分子瓦有三个粘性末端，记粘性末端 x 为表示值为 x 的序列，粘性末端 $\bar{x}$ 表示与粘性末端 x 互补的序列。右下角的粘性末端用于存储 p_i 的值，其他两个粘性末端用于二进制数 P 的分子瓦结构的内部连接。描述一个四位二进制数 P 的分子瓦结构如图 2(e)所示。它有 5 个分子瓦组成，其中 4 个为图 2(a)所示的分子瓦，另一个作为 P 结构结束的标志。图 2(b)所示的分子瓦结构用于描述二进制数 Q 的位。每个分子瓦有 4 个粘性末端，左右两个粘性末端用于二进制数 Q 的分子瓦结构的自身连接。正上方的粘性末端用于存储二进制数 N 的位，右上角的粘性末端存储 q_i 的值。描述一个四位二进制数 Q 的分子瓦结构如图 2(d)所示。它也有 5 个分子瓦组成。特殊的分子瓦 E_0 用于标记二进制数 Q 的分子瓦结构的结束。为连接 P 和 Q 的结构，我们设计了图 2(c)所示的分子瓦结构。表示 4 位二进制数 P 和 Q 的分子瓦结构连接示意图如图 2(f)所示。

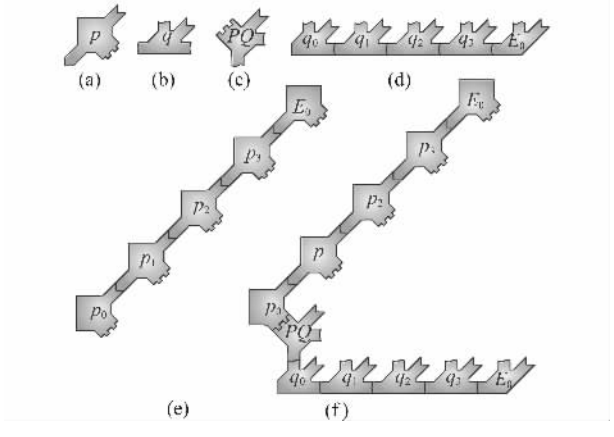

图 2 输入分子瓦和输入结构

表示二进制数 P 和 Q 的框架结构模拟了表 1 中常规乘法运算的输入模式。根据表 1 的计算规则，需要设计相应的计算分子瓦，用于计算二进制数 P 和 Q 的乘积。图 3 描述了计算中所需要的一些基本分子瓦类型，中间变量的值为该分子瓦的 $v(t)$ 值。图 3(a)是一种计算分子瓦类型，用来计算中间和，并传递相应的信息给其他分子瓦。它具有 6 个粘贴末端，其粘性末端序列分别表示变量 r、p、c 和 q，它们可以取值 0 或 1。左下角的三个粘性末端表示输入，右上角的三个粘性末端表示输出。在这个计算分子瓦中，基本的二进制操作位用变量 p 和 q 来描述，变量 r 和 c 分别用于描述中间结果与进位值。它们遵循如下规则

$$r' = (pq + c + r) \bmod 2,\ c' = (pq + c + r)/2$$

式中，r' 为新产生的中间结果；c' 为新产生的进位值。图 3(b)所示的分子瓦用于存储乘积 n_i，同时，它也传递 p_i 的值到相应的层。为避免非特异性杂交，不同类型分子瓦的粘性末端需要不同的序列，然而，有足够的序列可以得到。因此，我们可以采用这种方法计算较大数的乘积。

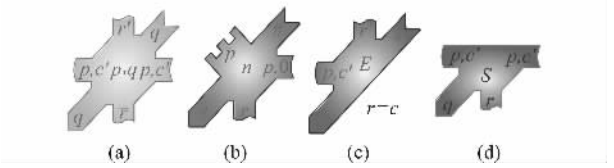

图 3 执行乘法运算所需要的基本 DNA 分子瓦

这里采用图 2 与图 3 所示的分子瓦来模拟表 1 的乘法运算规则。为更方便地读取结果，在 k 位二进制数 P 的末端填补 k 个 0。图 4(a)描述了用于两个 4 位数相乘的输入框架，其结果可能是 8 位数的情况。类似于表 1 的计算规则，计算分子瓦将自底向上组装到输入框架上。首先，标记为 r_{00} 的分子瓦组装到架子上(为便于描述，图 3(a)中的分子瓦用 r_{ij} 标记)，它计算 p_0 与 q_0 的乘积。接着标记为 n_0 和 $r_{0,1}$ 的分子瓦同时组装上去。标记为 n_0 的分子瓦存储 p_0 与 q_0 的乘积作为二进制数 N 的第一位，同时向右传递 p_1 的值。标记为 $r_{0,1}$ 的分子瓦计算 p_0 与 q_1 的乘积并存储于该分子瓦中。依次类推，完成整个计算过程。图 4(b)显示了一个自组装 DNA 计算执行乘法运算的组装示意图，每个分子瓦的右上角编号为该分子瓦的组装次序。最后，显示在图 3(c)中的分子瓦用于标记每一层的结束。

2.2 分解整数的非确定性算法

前面给出了一种采用 DNA 分子瓦自组装实现两个二进制数的乘法运算。从图 4(b)可以看出，给出二进制数 P 和 Q，我们可以得到相应的乘积，这里将根据已知的积去寻找相应的因子。借助于自组装 DNA 计算巨大并行性的特点，我们可以随机产生两个 k 位的整数 P 和 Q，利用前面的乘法运算方法，比较其乘积与给定的数是否相等。

作为组装的一部分，这里的二进制数 P 和 Q 将被动态创建。这表明，该计算时间相对于问题的规模是线性的。一旦有一对满足条件的二进制数 P 和 Q，计算中将被发现。为控制二进制数 P 和 Q 的位数，我们预先设定相应的输入框架。

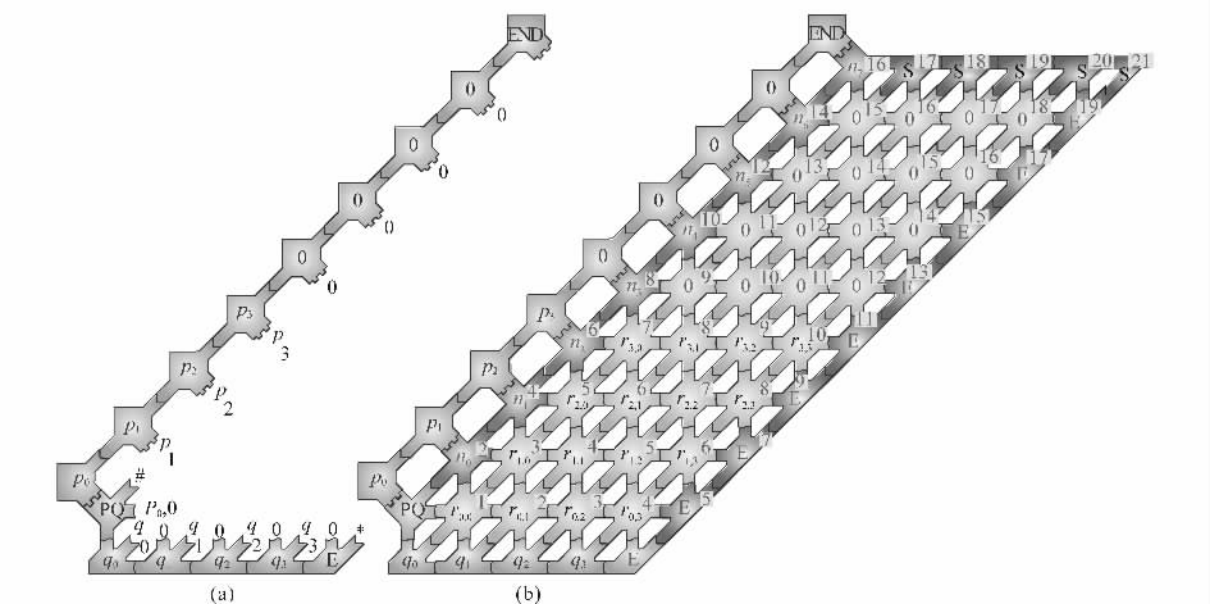

图 4　输入结构和执行乘法运算的自组装示意图

下面从算法层次上来描述该方法,它试图去模拟一个非确定性的计算机。非确定算法表明该算法中的一些步骤将存在一些非确定性的选择。指数级的自组装体期望能测试给定范围内所有可能的二进制数 P 和 Q,算法描述如下(其中第 4、5 步为非确定性的两步,用于随机指派 P 和 Q 的值):

Non-Determinstic (P,Q,N){
  $P=0$, $Q=0$;$R=0$; $p_0=q_0=1$
  for ($i=1$, …, $k-1$) {
    Assign a value (0 or 1) to variable p_i;
    Assign a value (0 or 1) to variable q_i;
    Computing $R=p_0\cdots p_i * q_0\cdots q_i$
    If ($n_i==r_i$) continue. //其中 r_i 为 R 的第 i 位。
    Else return failure. }
  Computing $R=P*Q$
  for ($i=k$, …, $2k-1$){
    If ($n_i==r_i$) continue.
    Else return failure.}
  Return and output Q.
  }

根据前面的乘法运算方案,该算法能直接借助于 DNA 分子瓦自组装实现。输入待分解的二进制数 N,用串联的 DNA 分子瓦编码(见图 5)。从图 5 可以看出,在最底端的输入层,有 k 个标记为'E'的分子瓦,用于连接指派 Q 值的分子瓦(图 6(a))。这一指派分子瓦的输出端不受输入端限制,其输入端均为 0 而输出端(q 的值)为 0 或 1,这体现了非确定性,也即选择的随机性。对于指派 P 值的分子瓦(图 6(b))也是如此,它的输出值 0 或 1 也不受输入信息的限制。

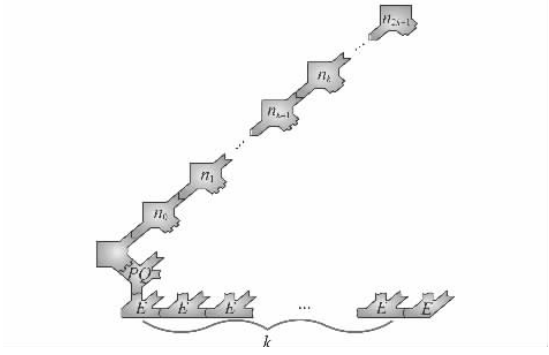

图 5　整数分解的输入结构

输入结构被复制数十亿次后放入已包含计算分子瓦(见图 6)的 DNA 溶液中,相应的分子瓦将组装到输入层。变量 p_i 和 q_i 将分别被随机指派值 0 或 1。用前面所提到的方法并行性计算所有可能的二进制数 P 和 Q 的乘积,每个组装体测试一种可能的二进制数 P 和 Q。一个输入结构和成功组装计算的实例如图 7 所示。计算结果借助于报道链读出(最上层将有一条单链通过表示二进制数 Q 的每个分子瓦,包含二进制数 Q 的每一位)。

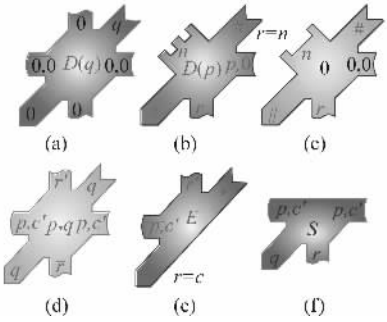

图 6　整数分解所需要的分子瓦

这里给出一个计算实例。我们考虑整数 $N=10001111_2$。图 7(a)给出了计算的初始状态,这时仅有一个分子瓦可以组装到这个输入结构上,它是一个指派分子瓦。现在我们向溶液中加入图 6 中描述的分子瓦,让它们一起退火,将得到最终的组装体。若整数 P 被指派为 1101_2,整数 Q 被指派为 1011_2。那么,$N=PQ$,最终组装体如图 7(b)所示。在这个结果层中,将有一条单链贯穿整个结果层,单链的开始处带有标记为'RES',这一单链包含了整数 Q。若整数 P 和 Q 不能被指派为整数 N 的因子,将导致计算失败。图 7(c)是一个不成功的例子,这里的数 Q 不是待分整数 N 的因子,导致没有相应的分子瓦组装到该组装体上。

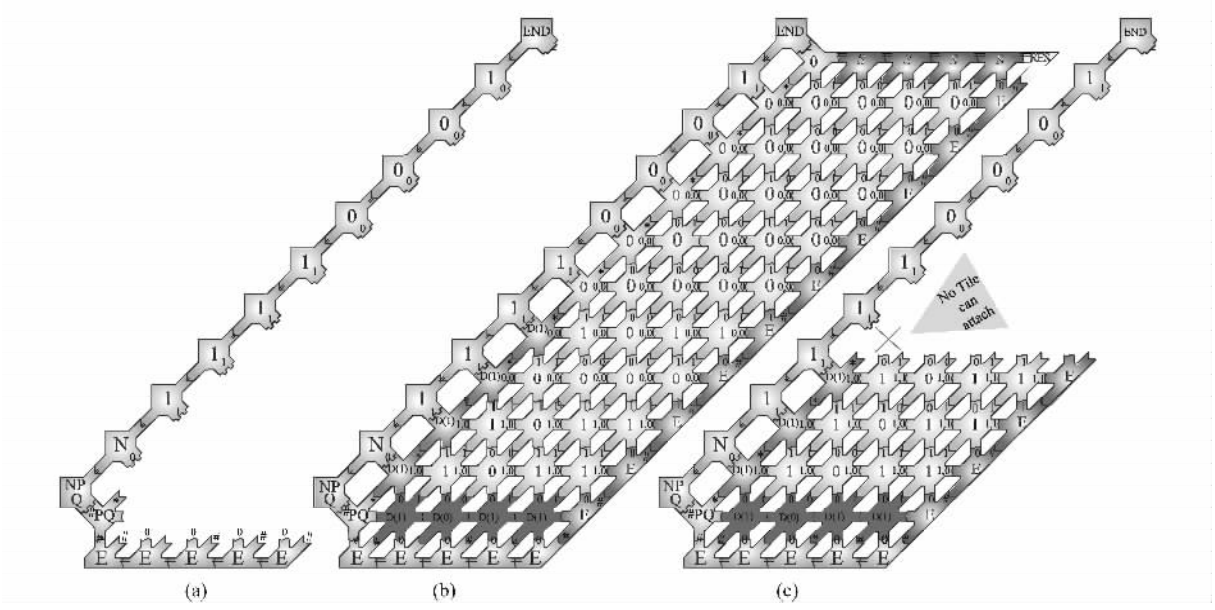

图 7 输入结构和分解整数 10001111_2 的实例

具体的生物实现,不再讨论。但是,本研究并未建立在任何的假设基础之上。

2.3 复杂度分析

这里根据计算所需要的时间、空间以及分子瓦类型数来分析系统的复杂度。从图 7 所示的例子中我们可以看出,其计算时间复杂度 T 也是相应组装体的深度。根据文献[20]中的时间复杂度计算理论,可以得出

$$T=(n+3)+(n+1)+3=2n+7=O(n^2)$$

系统的空间复杂度 S 即组装体的面积

$$S=(n+3)(n+2)+1=O(n)$$

式中,n 为被分解二进制数 N 的位数。

最后分析系统所需要的分子瓦类型数。

输入 由于变量 p 和 q 的取值均为二进制,因此表示变量 p 和 q 的分子瓦各需要 2 个。对于 P 和 Q 的输入结构,各需要 1 个输入边界分子瓦和一个连接输入结构 P 和 Q 的分子瓦。

计算 计算分子瓦共需 $2^4=16$ 个,用于执行乘法运算。另外将需要 3 个边界分子瓦。

输出 需要 4 个用于信息输出的分子瓦、一个输出边界分子瓦以及两个用于补 0 的分子瓦。

总共需要:$Sum=32=O(1)$。

3 结 论

鉴于目前破解 RSA 密码系统的困难性,本文在 Pelletier 等人基础上,引入非确定性算法,把 DNA 分子自组装技术应用到破解 RSA 密码系统中。提出了一种基于自组装 DNA 计算的 RSA 密码系统破译方案。虽然如此攻击的可行性最终将由实验条件来决定,但其成果是令人鼓舞的。

算法成功的关键在于,对于变量 p 或 $q\in\{0,1\}$,相应的指派分子瓦被指派每个值的机会必须是均等的,否则会由于需要的值得不到被指派的机会而导致算法失败。另一个是 DNA 计算自身的限制:问题的指数维度置换为实际空间被 DNA 分子瓦占据,这最终成为一种限制性的因素。分解整数的位数主要与分子瓦类型数和分子瓦总数有关。分子瓦类型数决定了所需 DNA 编码的长度,这里使用的分子瓦类型数是常数个,因此所需要的 DNA 序列编码长度对分解整数的位数影响较小。而由于分子瓦总数受计算空间复杂度限制,用于计算的 DNA 分子瓦数目并不能比 Avogadro 数目多(大约 10^{23}),因此,这种类型的攻击大概限制在 2^{80} 种不同的可能性。需要说明的是,实验条件对算法的可行性将起到非常重要的作用。同时,分子计算机的未来还是未知的,可能在将来分子计算机对于解决巨大的并

行计算问题是明确的选择,然而这在成为现实之前,还有许多技术问题有待解决,希望本文有助于论证分子瓦计算的技术价值。

参考文献:

[1] Adleman LM. Molecular computation of solutions to combinatorial problems[J]. *Science*, 1994,266(5187):1021 - 1024.

[2] Lipton R J. DNA solution of hard computational problems[J]. *Science*, 1995,268(5210):542 - 545.

[3] Adleman L M, Cheng Q, Goel A,et al. Combinatorial optimization problems in self-assembly[C]// *Proc. of the Annual ACM Symposium on Theory of Computing*, 2002:23 - 32.

[4] GehaniA, LaBean T, Reif H. DNA-based Cryptography[C]// *Proc. of the 5th DIMACS Workshop on DNA Based Computers*, 1999:233 - 249.

[5] Liu Q, Wang L, Frutos A G, et al. DNA computing on surfaces [J]. *Nature*, 2000,403(13):175 - 178.

[6] Winfree E. On the computational power of DNA annealing and ligation[C]// *DIMACS Series in Discrete Mathematics and Theoretical Computer Science*, 1996,27:199 - 211.

[7] Grunbaum B, Shephard G C. *Tilings and patterns*[M]. New York: Freeman,1986.

[8] Seeman N C. Biochemistry and structural DNA nanotechnology: an evolving symbiotic relationship[J]. *Biochemistry*, 2003,42 (24):7259 - 7269.

[9] Lipton R, Boneh D. Breaking DES using amolecular computer[C]// *DIMACS Series in Discrete Mathematics and Theoretical computer science*, 1996,27:37 - 66.

[10] Wang H. Proving theorems by pattern recognition I[J]. *Bell System Technical Journal*, 1961,40:1 - 42.

[11] Seeman N C. Nucleic acid junctions and lattices[J]. *Journal of Theoretical Biology*, 1982,99(2):237 - 247.

[12] Seeman N C, Chen J H, Kallenbach N R. Gel electrophoretic analysis of DNA branched junctions [J]. *Electrophoresis*, 1989,10:345 - 354.

[13] Winfree E, Liu F, Wenzler L,et al. Design and self-assembly of two-dimensional DNA crystals[J]. *Nature*,1998,394:539 - 544.

[14] Li X, Yang X, Qi J, et al. Antiparallel DNA double crossover molecules as components for nanoconstruction[J]. *Journal of the American Chemical Society*, 1996,118:6131 - 6140.

[15] LaBean T, Yan H, Kopatsch J, et al. The construction, analysis, ligation and self-assembly of DNA triple crossover complexes[J]. *Journal of the American Chemical Society*, 2000,122: 1848 - 1860.

[16] Liu D, Park S, Reif J, et al. DNA nanotubes self-assembled from triple-crossover tiles as templates for conductive nanowires[J]. *Proc. of the National Academy of Sciences of the United States of America*, 2004,101:717 - 722.

[17] Mao C, Sun W,Seeman N C. Designed two-dimensional DNA Holliday junction arrays visualized by atomic force microscopy[J]. *Journal of the American Chemical Society*, 1999, 121: 5437 -5443.

[18] Adleman L. Toward a mathematical theory of self-assembly [R]. USC,2000.

[19] Rothemund W K, Papadakis N, Winfree E. Algorithmic self-assembly of DNA Sierpinski triangles[J]. *PLoS Biol*, 2004,2 (12):e424.

[20] Brun Y. Arithmetic computation in the tile assembly model: Addition and multiplication[J]. *Theoretical Computer Science*, 2006,378(1):17 - 31.

[21] Brun Y. Nondeterministic polynomial time factoring in the tile assembly model[J]. *Theoretical Computer Science*, 2008,395 (1):3 - 23.

[22] Brun Y. Solving NP-complete problems in the tile assembly model[J]. *Theoretical Computer Science*, 2008,395(1):31 - 46.

[23] Zhang X C, Wang Y F, Chen Z H,et al. Arithmetic computation using self-assembly of DNA tiles: subtraction and division [J]. *Progress in Natural Science*, 2009,19(3):377 - 388.

[24] Reif J, LaBean T, Seeman N. Challenges and applications for self-assembled DNA nanostructures[C]// *Proc. of the Sixth International Workshop on DNA-Based Computers*, 2001:173 - 198.

[25] Chang W L, Guo M, Michael H. Fast parallel molecular algorithms for DNA-based computation [J]. *IEEE Trans. on Nanobioscience*, 2005,4(2):133 - 163.

[26] Beaver D. Factoring:the DNA solution[C]// *Proc. of the 4th International Conference on the Theory and Applications of Cryptology: Advances in Cryptology*, 1994:419 - 423.

[27] Parhami B. *Computer arithmetic: algorithms and hardware designs*[M]. New York: Oxford University Press,2000.

[28] Pelletier O, Weimerskirch A. Algorithmic self-assembly of DNA tiles and its application to cryptanalysis[C]// *Proc. of the Genetic and Evolutionary Computation Conference*, 2002: 139 - 146.