DOI:10.7544/issn1000-1239.2016.20150312
53(11):2583-2593,2016

DNA 自组装计算模型求解二部图完美匹配问题

蓝雯飞　邢志宝　黄　俊　强小利
(中南民族大学计算机科学学院　武汉　430074)
(lanwenfei1@163.com)

The DNA Self-Assembly Computing Model for Solving Perfect Matching Problem of Bipartite Graph

Lan Wenfei, Xing Zhibao, Huang Jun, and Qiang Xiaoli
(*College of Computer Science, South-Central University for Nationalities, Wuhan* 430074)

Abstract　Biological systems are far more complex and robust than systems we can engineer today, and Because of its advantages of stability and specificity, DNA molecules have been used for the construction of nano-scale structures. With the development of DNA molecule self-assembly strategy, lots of programmable DNA tiles are constructed and used for solving NP problems. The tile assembly model, a formal model of crystal growth, is a highly distributed parallel model of nature's self-assembly with the traits of highly distributed parallel, massive storage density and low power consumption. In the system, a function can be computed by deterministic assembly and identified two important quantities: the number of tile types and the assembly speed of the computation. Here a DNA self-assembly model for perfect matching problem of bipartite graph is demonstrated, and a 10-vertices bipartite graph is used as an example to illustrate this model. The computation is nondeterministic and each parallel assembly is executed in time linear in the input. The result shows that the successful solutions can be found among the many parallel assemblies, and it requires only a constant number of different tile types: 14.

Key words　perfect matching; bipartite graph; DNA computing; self-assembly; tile

摘　要　针对二部图完美匹配问题,提出了一种基于 DNA 计算自组装模型的算法.首先,通过该算法求解了一个具有 10 个顶点的二部图完美匹配问题的实例,实例中给出 DNA 计算自组装模型算法所涉及到的 DNA Tile 的编码设计方案、自组装计算步骤及结果分析;然后,给出了任意二部图完美匹配问题的求解方案;最后,针对 DNA 计算自组装模型算法解决任意二部图完美匹配问题的时间和空间消耗进行了讨论.结果表明:对任意二部图只需 14 种 Tile 类型就能够得到完美匹配.

关键词　完美匹配;二部图;DNA 计算;自组装;瓦片

中图法分类号　TP301.6

收稿日期:2015-04-22;**修回日期**:2016-04-28

基金项目:国家自然科学基金项目(61379059);中央高校基本科研业务费专项基金项目(CZZ13003);2015 年中南民族大学研究生优秀学位论文培育项目

This work was supported by the National Natural Science Foundation of China (61379059), the Fundamental Research Funds for the Central Universities (CZZ13003), and the Master Candidate Excellent Thesis Cultivation Project of South-Central University for Nationalities in 2015.

通信作者:强小利(qiangxl@mail.scuec.edu.cn)

在计算面临的三大类问题(即容易、困难和不可计算)中,对于容易问题,传统电子计算机可以轻松地处理,但是对于困难问题(一般指 NP-完全问题),计算时间将会随着计算规模的增大而呈指数增长.针对这个问题,很多科学家将目光转向对其他类型的计算机结构研究上,从而类似于神经网络计算机[1-2]、量子计算机[3-4]以及 DNA 计算机模型[5-11]等应运而生. DNA 计算机模型,凭借其高度的并行计算能力和海量的存储能力在各种新型计算机中脱颖而出,在很多领域(比如生物、物理、数学及计算机科学)中都得到了广泛的应用,并成为解决 NP 难问题的一种潜在的方案.

早在 20 世纪 60 年代,Feynman[12]就已经提出了在分子水平上进行计算的概念,但是这种观念真正引起各个领域科学家的兴趣是在 Adleman[13]提出利用 DNA 计算解决有向 Hamilton 路径问题之后. Aldeman 不但解决了 Hamilton 路径问题,还成功地利用现代生物技术进行了实验.自此,DNA 计算引起各领域科学家的极大兴趣以及社会各界的广泛的关注和支持. 1998 年,Winfree[14]提出了一种既具有 Seeman 的分支 DNA 结构又具有 Wang Tiling 理论的可以自组装的 DNA 结构. 2009 年张勋才[15]通过编码 Tile 粘性末端,将待运算的信息与 Tile 的结合域相结合,建立了基于 DNA 计算的减法和除法运算模型,其中除法模型中包含了比较、复制和减法 3 个子系统.作者利用 DAN 计算模型实现了这 3 个子系统并将其合并最终成为了一个完整的除法模型.此外,还利用自组装 DNA 计算模型解决了 0-1 规划问题和图着色问题.在解决 0-1 规划问题时,作者将 0-1 规划问题中的约束机制分成了“与”操作和“比较”操作 2 个基本操作,并利用 DNA 自组装模型实现了这 2 种操作.通过这 2 种操作的组合,可以自动判断任意可行解是否满足给定的约束条件.在图着色上,作者引入了非确定性算法,利用 DNA 自组装的高度并行的特性,验证所有可能的着色方案,在多项式时间内就可以解决图着色问题. 2009 年,陈智华[16]利用 DNA 自组装模型提出了一次一密密码系统.该系统由 4 个子系统组成:加密子系统、密码提取子系统、秘钥计算子系统和解密子系统,同时利用生物技术实现了秘钥的安全传输.另外,通过对 DNA Tile 的编码实现了破译 Diffie-Hellman 算法的 DNA 自组装模型.通过生物技术(PCR 和凝胶电泳)读出素数原根的离散对数值,从而威胁 Diffie-Hellman 密钥交换安全.但是由于 Tile 的种类与输入位数呈线性关系,限制了该方案对大规模输入的应用. 2008 年,Brun[17]基于 DNA 自组装计算模型提出了 2 个系统,解决了 NP-完全问题——k-SAT,并以 3-SAT 为例进行了说明.作者提出的第 1 个 Tile 系统用了 $\Theta(n^2)$种不同的 Tile 表示布尔方程式中 n 个不同的变量;第 2 个 Tile 系统仅用 $64=\Theta(1)$ 种不同的 Tile 表示具有 n 个变量的布尔方程式,大大降低了自组装运算的空间消耗.运算时,根据 Tile 的编码及运算规则随机形成解空间,再通过解空间中的每个解对布尔方程中的每个变量进行扫描,确定解空间的正确性.同年,他又用 49 种不同的 Tile,建立了用于求解子集和问题的 DNA 自组装计算模型[18].

二部图匹配问题是图论中的热点问题,在解决各个领域的问题中也得到了广泛的应用.在二部图匹配问题中,除了匈牙利算法之外,还有分层网络算法[19]、最大流算法以及基于量子协同的智能优化算法[20]等.在 DNA 计算产生之后, 2002 年高琳等人[21]提出了一种基于质粒 DNA 的计算模型,并将其用于求解二部图的匹配问题.同年 Wang[22]提出了一种用于求解二部图最大匹配的 DNA 算法. 2003 年董亚非[23]利用肽核酸(PNA)提出粘贴模型的完美匹配问题的分子算法.该算法首先将问题映射到 DNA 存储链,利用 PNA 分子在无盐状态下比 DNA 分子更容易与 DNA 分子结合并稳定存在的特性,设计编码 DNA 存储链的补链(PNA 粘贴链),以达到计算的目的. 2011 年,周旭等人[24]给出了一种最大匹配问题的 DNA 计算算法,该方法在保持多项式操作时间的条件下,将解空间从 $O(2^m)$减少到 $O(1.618^m)$.

本文则是利用 DNA 自组装模型的高度并行性,结合二部图本身的特征,建立了一种用于求解二部图完美匹配的一种自组装计算模型,对任意二部图只需 14 种 Tile 类型就能够得到完美匹配.

1 DNA 计算自组装模型

文献[14-16]中对自组装模型的定义及相关符号描述如下:

定义 1. 一个 DNA 计算自组装系统 S 是一个 3 元组$\langle T,g,\tau\rangle$,其中 T 为 DNA Tile 集合,g 为结合强度函数,τ 为热力学温度.

DNA Tile 是带有 4 个粘性末端的分子元件,它可以和带有与其互补的粘性末端的 DNA Tile 相结

合,形成规模庞大的带有唯一结果的二维结构. 一个 Tile 是一个四元组$\langle\sigma_N,\sigma_E,\sigma_S,\sigma_W\rangle\in\Sigma^4$,这里 Σ 是具有粘性末端集合. Σ 是一个有限的字母(结合域)集合,包括空集 null,如果没有特别指出,$empty=\langle$null,null,null,null$\rangle$是一个特殊的 Tile,表示不与任何 Tile 相结合的 Tile.

结合强度函数 $g:\Sigma\times\Sigma\to\mathbb{R}$, $\forall\sigma\in\Sigma, g(\text{null},\sigma)=0$,表示粘贴域强度为 0.

对于方向集 $D=\{N,E,S,W\}$来讲,对于所有点的坐标$(x,y)\in\mathbb{Z}^2$,都有 $N(x,y)=(x,y+1)$, $E(x,y)=(x+1,y)$, $S(x,y)=(x,y-1)$,和 $W(x,y)=(x-1,y)$. 如果$\exists d\in D$,使 $d(x,y)=(x',y')$,那么就说位置(x,y)和位置(x',y')相邻. 对于一个 Tile t, $d\in D$,则用 $bd_d(t)$表示 Tile t 在边 d 上的结合域.

若 T 是一个包含 *empty* Tile 的一个集合, $A:\mathbb{Z}\times\mathbb{Z}\to T$ 是 T 的一个配置. A 是有限的,即 $A(x,y)\neq empty,(x,y)\in T$. $A(x,y)=t$ 表示 Tile t 在位置(x,y)处与配置 A 结合.

定义 2. 在一个 Tile 系统中,A 是部分 Tile 集 $T\subseteq\Sigma^4$ 的一个配置,那么要使 Tile $t\in T$ 能够在位置(x,y)上结合到 A 上,从而产生一个新的配置 A',必须满足 4 个条件:

1) $(x,y)\in A$;

2) $\Sigma_{d\in D}g(bd_d(t),bd_{d^{-1}}(A(d(x,y))))>\tau$ 且 d^{-1}表示和 d 在方向上对应的边;

3) $\forall(u,v)\in\mathbb{Z}^2,(u,v)\neq(x,y)\Rightarrow A'(u,v)=A(u,v)$;

4) $A(x,y)=t$.

即要使一个 Tile 能够稳定结合到一个配置上,当且仅当它与相邻的 Tile 的结合域强度之和大于或等于热力学温度 τ. 对所有结合域 $\sigma,g(\sigma,\sigma)=1$, $\tau=2$,那么一个 Tile t 要想结合到某位置上,它至少要和 2 个 Tile 结合.

定义 3. 在一个 Tile 系统 S 中,A 是 Tile 集 $T'\subseteq\Sigma^4$ 的一个配置,那么一个 Tile $t\in T$ 能在位置(x,y)上从配置 A 上分离下来并产生一个新的配置 A',当且仅当其满足 4 个条件:

1) $(x,y)\in A$;

2) $\Sigma_{d\in D}g(bd_d(t),bd_{d^{-1}}(A(d(x,y))))<\tau$;

3) $\forall(u,v)\in\mathbb{Z}^2,(u,v)\neq(x,y)\Rightarrow A'(u,v)=A(u,v)$;

4) $(x,y)\notin A'$.

即当一个 Tile 和与它相邻 Tile 的结合域强度之和小于热力学温度 τ,它就能从所在配置上分离下来. 例如,对所有的 $\sigma,g(\sigma,\sigma)=1$ 并且 $\tau=2$,如果少于 2 个 Tile 在与 Tile t 相邻的位置与之结合,那么 Tile t 就会从其位置上分离下来.

经过连续不断的组装后得到的最后结果,称为最终配置. 如果在所有操作下得到的最终配置相同,则称系统产生唯一的最终配置.

2 求解完美匹配问题的 DNA 自组装计算模型

2.1 完美匹配问题

在无向图 $G=(V,E)$中,边集 E 的任意一个子集 $M\subseteq E$,若 M 中的任意 2 条边都没有公共端点,则称 M 为图 G 的一个匹配. 若匹配 M 中的所有与匹配边相关联的顶点称为关于 M 饱和点,否则称为 M 非饱和点[24].

M 是二部图 G 中任意一个匹配,若 G 的任意一个互补点集 V_1 和 V_2 中的所有点都是 M 饱和点,则 M 就是一个完美匹配[24-26]. 如图 1 所示,G 是一个具有 10 个顶点的二部图. 其中 $M_1=\{e_1,e_4,e_8,e_{11},e_{12}\}$, $M_2=\{e_1,e_5,e_7,e_{11},e_{12}\}$都是图 G 的完美匹配.

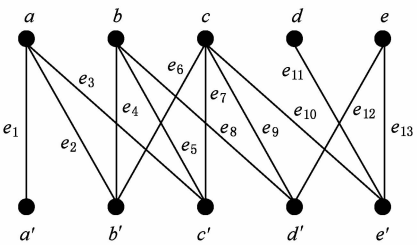


Fig. 1 Bipartite graph G.

图 1 二部图 G

2.2 算法模型实例

本文结合完美匹配的概念及图中各顶点间的邻接关系,给出了一个基于 DNA 计算自组装模型的系统,并为该系统设计编码 DNA Tile 以及种子配置. 在这个系统中,DAN Tile 在一定的控制条件下执行自组装运算,整个运算过程从种子配置开始自右向左、自下向上进行,最终将会计算出一个组装体. 若自组装得到的结果是完美匹配,将会在模块的左上角出现一个 *Success* 标志的 DNA Tile,否则模块将会在下一步的升温过程中溶解掉.

本文将这个系统定义为完美匹配自组装系统 $S_{\text{pm}}=\langle T_{\text{pm}},g_{\text{pm}},\tau_{\text{pm}}\rangle$,其中:$T_{\text{pm}}$为自组装模型获得完美匹配所使用的所有 Tile 的集合;g_{pm}是 S_{pm}的结合

函数,S_{pm}的结合域为 $\Sigma_{pm}=\{null,\chi,r,-,=,v\}$,$\chi=\{a,b,c,d,e,a',b',c',d',e'\}$. 结合函数定义如下:

1) $\forall\sigma\in\Sigma_{pm}, g_{pm}(\sigma,null)=g_{pm}(null,\sigma)=0$;

2) $\forall\sigma\in\Sigma_{pm}\backslash\{null,=,v\}, g_{pm}(\sigma,\sigma)=1$;

3) $g_{pm}(v,v)=2$;

4) $g_{pm}(=,=)=2$;

5) $\forall\sigma,\sigma'\in\Sigma_{pm}$, if $\sigma\neq\sigma'$, then $g_{pm}(\sigma,\sigma')=0$.

任意结合域与 null 的结合强度为 0. 除结合域 $=,v$ 外,其余结合域与其自身的结合强度都为 1. $=,v$ 与其自身结合强度为 2. 任何结合域与非自身结合域的结合强度为 0. 设定系统温度 $\tau=2$,也就是说只有当一个 Tile 的结合域强度之和大于或等于 2 时,它才能稳定地结合到已有的集合上.

2.3 DNA Tile 编码设计

根据上述算法及完美匹配概念,本文对完美匹配自组装系统中的 DNA Tile 的编码设计如图 2 所示:

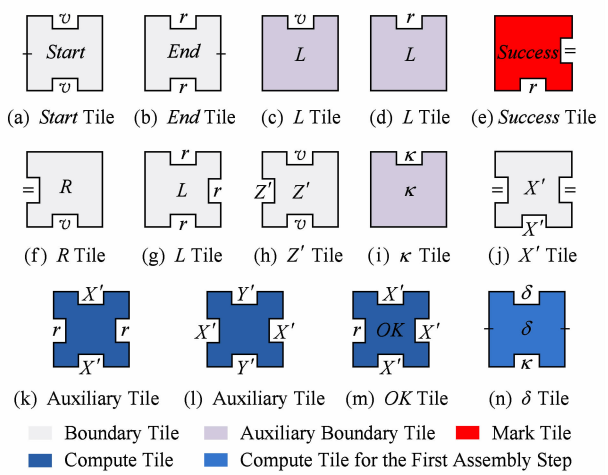


Fig. 2 Compute Tile.

图 2 运算 Tile

图 2(a)~(g)所示为完美匹配自组装系统 S_{pm} 的所有组装都必须用到边界 Tile.

图 2(a)中的 *Start* Tile 是组装开始的标志,自组装计算从 *Start* Tile 开始向左执行. 由图 2(a)可知 $Start=\langle v,null,v,-\rangle$,即只有满足 $bd_S(t)=v$,$bd_N(t)=v$,或 $bd_E(t)=-$(其中 $-$ 表示 Tile 的结合域)的 Tile 才能与之结合. 图 2(b)中的 Tile 记为 *End* Tile 是第 1 行组装完成的标志,它控制自组装计算的方向,使其只能向上增长. $End=\langle r,-,r,null\rangle$,只有满足 $bd_S(t)=r$,$bd_N(t)=r$,或 $bd_W(t)=-$ 的 Tile 才能与之结合. 图 2(c)和图 2(d)中的 Tile 都是辅助 Tile,都在种子配置第 0 行. 前者位于最右端,其北边与 *Start* Tile 结合,后者位于最左端,与 *End* Tile 相结合. 图 2(e)中的 Tile 记为 *Success* Tile,是自组装结束的标志,也是组装成功的标志. 当自组装得到的是正确的完美匹配时,将会在组装体的左上角出现一个成功标志,即 *Success* Tile. 由图 2 可知,$Success=\langle null,=,r,null\rangle$,$bd_N(R)=null$,$bd_E(R)==$(第 2 个=表示结合域),$bd_S(R)=r$,$bd_W(R)=null$,其西边和北边的结合域均为 null,因此它可使得自组装运算不能继续向上和向左执行,从而结束整个组装过程.

图 2(f)中的 Tile 记为 *R* Tile,是自组装最后一步的开始标志,它只能在 *S* 向与 Z' Tile(图 2(h))相结合. 由于 $bd_N(R)=null$,所以 *R* Tile 的北边不能与任何 Tile 相结合,从而控制整个组织运算过程不能继续向上执行. 图 2(g)中的 Tile 记为 *L* Tile,$L=\langle r,r,r,null\rangle$,是标志每行自组装运算结束的 Tile. 由于 $bd_W(L)=null$,所以没有 Tile 能够在其西边与之结合,从而控制每行自组装运算使其不能继续向左执行.

本文对完美匹配自组装系统 S_{pm} 中运算 Tile 的编码设计如图 2(h)~(n)所示. 其中,图 2(h)中的 Z' Tile 是控制每行自组装运算向左执行的右边界运算 Tile,它的 *S* 和 *N* 可与其他 Z' Tile 或 *R* Tile 相结合构成种子配置的第 0 列. 图 2(i)中的 Tile 记为 $\kappa=\langle\kappa,null,null,null\rangle$,是按顶点度数大小从右向左排列形成种子系统第 0 行的 Tile,它的 *N* 与图 2(n)中满足条件的运算 Tile 的 *S* 相结合. 图 2(j)中的 X' Tile 是上边界 Tile,是标志每一列自组装运算终止的 Tile,也是用于读取完美匹配结果的 Tile. 其中,X' Tile 体上的 X' 是指当前列的顶点信息.

图 2(k)中 Tile $\langle X',r,X',r,\rangle$的作用是传递信息. X'将当前列的顶点信息向上传递,r 将前面已经出现 *OK* Tile(图 2(m))的信息向左传递. 图 2(l)中的 Tile $\langle Y',X',Y',X'\rangle$,当 $bd_S(t)\neq bd_E(t)$时,X' 和 Y'分别将当前行和当前列的顶点信息向左和向上传递. 图 2(m)中的 Tile $OK=\langle X',X',X',r\rangle$,当 $bd_S(t)=bd_E(t)$时,西边的 r 将已有 *OK* Tile 的信息向左传递,同时其北边 X' 将当前列的顶点信息向上传递. 图 2(n)所示 Tile $\delta=\langle\delta,-,\kappa,-\rangle$. 对 $\forall i\in1,2,\cdots,n/2,\kappa_i,\delta_i\in\chi$,其中,$\kappa_i$ 是种子系统第 0 行第 i 列的 Tile 信息,枚举了种子系统第 0 行中所有的 Tile 的信息(种子系统中所有 Tile 的 *N* 的结合域). δ_i 枚举了所有能与 κ_i Tile 相结合的 Tile 的

信息.κ 是由 κ_i 组成的集合,δ 是由 δ_i 组成的集合.不同的 δ Tile 间可以在 E 和 W 相互结合.

2.4 运算 Tile 的设计算法

根据 2.1 节可知,二部图的完美匹配作为一种特殊的匹配方式,它具有 3 个基本要求:1)必须包含图中所有顶点;2)完美匹配中的任意 2 条边都不能有公共顶点(任意 2 条边都不相邻);3)完美匹配中每条边的 2 个端点必须分别属于二部图的 2 个互补点集.根据以上 3 点可知,若图中存在 1 度顶点,那么与这个 1 度顶点相关联的边有且只有这一条包含在完美匹配中.所以为了简化求解的复杂性,本文在编码设计 DNA Tile 前,首先找出所有子图中的 1 度顶点、1 度顶点的邻接点及与它们相关联的边,并根据这种关系设计编码对应的顶点 Tile.具体算法如 $BigraphToTile(G,E,V)$所示,其中 V 和 E 是图 G 的顶点集和边集.

算法 1. $BigraphToTile(G)$.

$S\leftarrow$正偶数集合;
$G=(V,E)$;
$TILE\leftarrow\varnothing$;
if $|V|\in S$
  while $(V'\neq\varnothing)$
    if $v\in V'$
      $TILE\leftarrow TILE\cup\{T_v,T_{v^{-1}}\}$;
      design T_v 和 $T_{v^{-1}}$ 使得 $bd_N(v)=bd_S(v^{-1})$;
      $E\leftarrow E-\{E_v,E_{v^{-1}}\}$;
      $V\leftarrow V-\{v,v^{-1}\}$;
    end if
    $G=(V,E)$;
    $BigraphToTile(G)$;
  end while
  while $(V'=\varnothing$且$V(G)=\varnothing)$
    if $v\in V$ 且 $v\neq\varnothing$
      design T_{v_i} 和 $T_{v_i^{-1}}$ 使得 $bd_N(v_i)=bd_S(v_i^{-1})$;
      $TILE\leftarrow TILE\cup\{T_{v_i},T_{v_i^{-1}}\}$;
    end if
  end while
end if
return $TILE$.

算法 1 中,G 表示代求解的图,V 和 E 分别为 G 的顶点集合和边集;V' 表示图 G 中所有 1 度顶点构成的集合,v 表示图 G 中的顶点,v^{-1} 为 v 相邻顶点;T_v表示所设计的关于顶点 v 的所有 Tile 构成的集合,$T_{v^{-1}}$ 是关于顶点 v^{-1} 所设计的所有 Tile 构成的集合,E_v 表示与顶点 v 相关联的边集.$TILE$ 是所有 1 度顶点和非 1 度顶点的顶点 Tile 的集合.

如算法 1 所示,本文的 DNA Tile 编码设计步骤如下:首先找出原图 G 中 1 度顶点 v 构成的集合 V',若 V' 中存在 1 度顶点,那么设计 DNA Tile 使 v Tile 只能与 v^{-1} Tile 相结合;从原图删除顶点 v 和 v^{-1},重置图 G;重复上述操作,直到图 G 中顶点为 0.若 $V'=\varnothing$,即图中没有 1 度顶点,则为 G 中的每对相邻的顶点设计顶点 Tile,使 G 中每对相邻的顶点所对应 Tile 可以相互结合.

对于一个具有 10 个顶点的二部图(如图 1 所示),其完美匹配自组装系统的种子配置如图 3 所示:

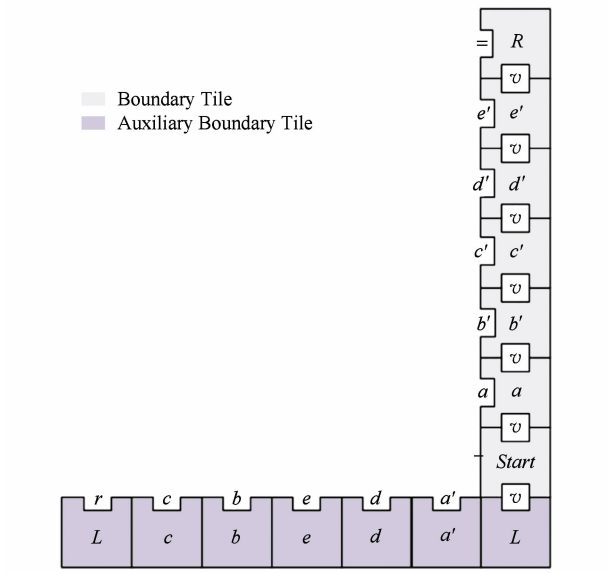


Fig. 3 The seed configuration of the graph G shown in Fig. 1.

图 3 图 1 中图 G 的完美匹配自组装系统的种子配置

根据算法 1,将 5 个顶点 Tile 按度数从小到大、从右向左排列,形成种子配置的第 0 行.剩下的 5 个顶点 Tile 构成自组装运算所需的 Tile 集 T.同时将 T 中的所有 Tile 通过 S 和 N 的相互结合,由下至上排列形成种子系统第 0 列.

在 $\tau=2$ 的条件下,按照编码规则,在自组装的第 1 步结合上去的 $t\in T$ 将会形成一个计算结果,本文将其称为结果序列.根据 DNA 自组装计算的高度并行性及海量的存储能力可知,若自组装运算数量足够大,最终一定能够得到完美匹配.但由于结合的随机性,结果序列中可能会出现重复的顶点 Tile,很显然这种结果序列不是完美匹配.

为了将完美匹配从众多的结果序列中分离出来,本文同时在种子配置中设计了结果序列检测部分,

检测结果序列中是否有重复的顶点 Tile. 若没有，说明结果序列是完美匹配，将在组装体左上角出现一个 *Success* Tile，从而分离出完美匹配；否则，组装体将在下一步升温过程中溶解. T 中所有 Tile 的 S 和 N 相互结合，形成种子配置的第 0 列，即种子配置的结果序列检测部分. 这里称结果序列检测部分的每个 Tile 叫检测 Tile.

2.5 算法步骤

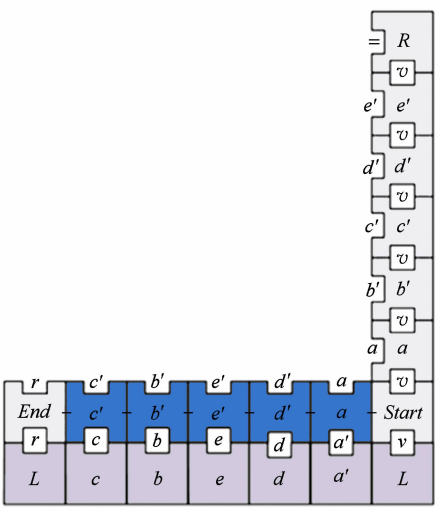
以图 1 中的二部图 G 为例，讨论完美匹配自组装的算法步骤. 图 G 有 10 个顶点，它的完美匹配自组装系统的种子配置 S'_{pm}，如图 3 所示. *Start* Tile 所在位置为 (x_s, y_s)，那么：

$$bd_W(S'_{\text{pm}}(x_s, y_s+i))=Z_i, Z_i\in\{a, b', c', d', e'\},$$
$$bd_N(S'_{\text{pm}}(x_s-i, y_s-1))=\kappa_i, \kappa_i\in\{a', b, c, d, e\},$$
$$bd_N(S'_{\text{pm}}(x_s-6, y_s-1))=r,$$
$$bd_W(S'_{\text{pm}}(x_s, y_s+6))==.$$

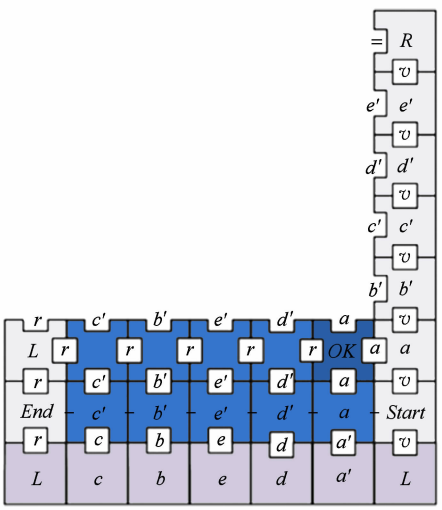
这里 $i\in\{1, 2, \cdots, 5\}$. 对于所有其他位置 (x, y)，$S'_{\text{pm}}(x, y)=empty$. 在自组装过程中，每一步组装到配置上的 Tile 与原配置一起形成一个新的配置. 本文将第 t 步自组装完成后形成的新配置记作 S_t，$S_0=S'_{\text{pm}}$，使 $Q_t\subseteq\mathbb{Z}^2$ 是能在第 t 步结合到配置 S_{t-1} 上的所有位置的集合. $\forall t\in\mathbb{N}_+$，对所有的位置 $W_t\in Q_t$，U_t 是能在位置 W_t 处结合到 S_{t-1} 上的 Tile 集.

在 $\tau=2$ 的条件下，二部图 G 的完美匹配自组装过程如下：

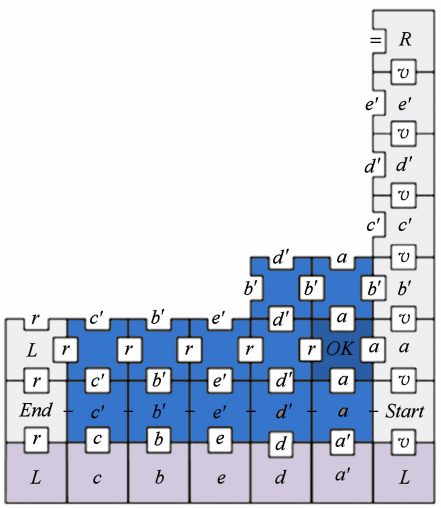
Step1. $\forall i=0, 1, \cdots, 4$，位置 $(x_s-i, y_s)\in Q_1$，$bd_W(S_0(x_s-i, y_s))=-$，$bd_N(S_0(x_s-i-1, y_s-1))=\kappa_i$. 只有满足结合域为 $bd_E(S_0(x_s-i, y_s))=-$，$bd_S(S_0(x_s-i, y_s))=\kappa_i$ 的 Tile t，$t\in U_1$，才能稳定地结合到 S_0 上. 在位置 $(x_s-6, y_s)\in Q_1$，$bd_W(S_0(x_s-5, y_s))=-$，$bd_N(S_0(x_s-6, y_s-1))=r$，$U_1$ 中只有结合域 $bd_S(t)=r$，$bd_E(t)=r$ 的 *End* Tile 满足条件. *End* Tile 结合到配置上形成新的配置 S_1. Step1 完成后，所产生的组装体如图 4(a)所示.



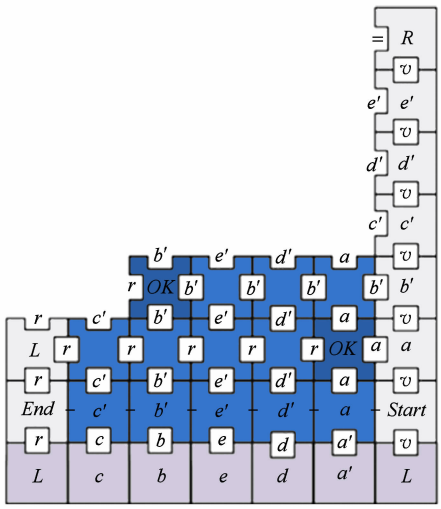
(a) The result of Step 1



(b) Detect by using a Tile



(c) The binding domain on W and N directions



(d) Detect by using b' Tile

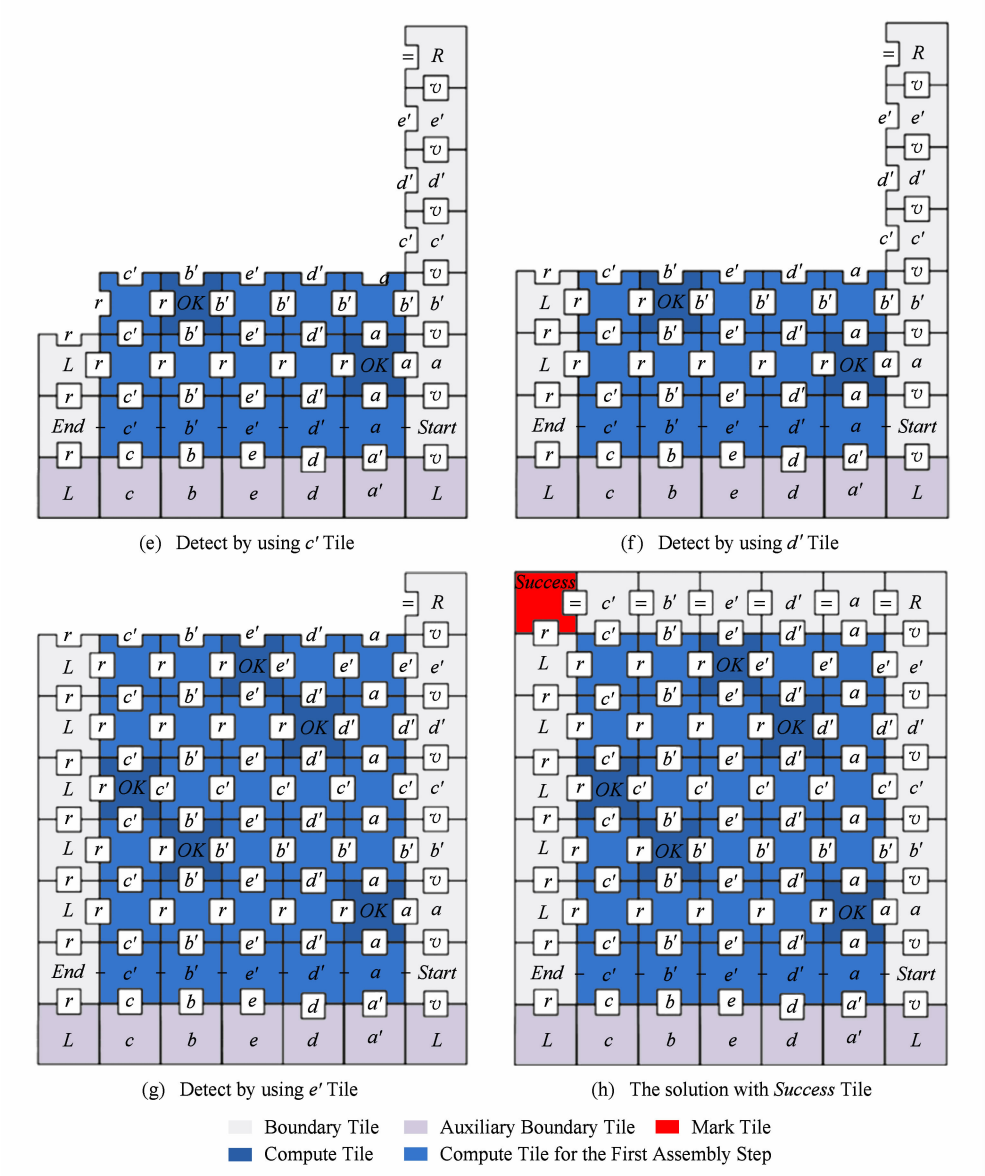


Fig. 4 Perfect matching self-assembly model of the bipartite graph G shown in Fig. 1.

图 4 图 1 二部图 G 的完美匹配自组装模型

Step2. Step1 组装完成后得到一个结果序列. 如 2.4 节所述,该结果序列中可能存在重复的顶点 Tile, 需要对这个结果序列进行检测. 如图 4(a)所示,S'_{pm}的第 1 个检测 Tile 是 a Tile. 在此步骤中,$\forall i=1,2,\cdots,5$,a Tile 向左扫描结果序列的每个 Tile $S_1(x_s-i,y_s)$,检测在位置$(x_s-i,y_s+1)\in Q_2$处,$bd_N(S_1(x_s-i,y_s))=bd_W(S_1(x_s-i+1,y_s+1))=a$ 是否成立. 若不成立,则将 $bd_W(S_1(x_s-i+1,y_s+1))=a$ 向左传递使 $bd_W(S_1(x_s-i,y_s+1))=a$;若成立,则将相等信息 r 向左传递使 $bd_W(S_1(x_s-i,y_s+1))=r$,将 $bd_N(S_1(x_s-i,y_s))=\kappa_i$, 向上传递使 $bd_N(S_1(x_s-i,y_s+1))=\kappa_i$.

在此过程中,a Tile 一旦在结果序列中扫描到自身,将不会继续扫描,而是将信号 r 继续向左传递. 在位置(x_s-6,y_s+1)上,因为 $bd_W(S_1(x_s-5,y_s+1))=r$,$bd_N(S_1(x_s-6,y_s))=r$,故在 $t\in U_2$ 中,只有满足结合域为 $bd_S(t)=r$ 和 $bd_E(t)=r$ 的 L Tile 才能稳定地结合到配置上,形成配置 S_2,如图 4(b)所示.

Step3. 重复 Step2,分别对对应位置上 b' Tile, c' Tile, d' Tile 和 e' Tile 进行检测. 检测结果如图 4(c)~(f)所示. 在检测完 e' Tile 之后,形成配置 S_6,如图 4(g)所示.

Step4. 完成自组装. 由图 4(g)可知,$\forall i=1,2,\cdots,5$,配置 S_6 中的位置$(x_s-i,y_s+6)\in Q_7$,暴露在外面的结合域为 $bd_W(S_6(x_s-i+1,y_s+6))==$,

$bd_N(S_6(x_s-i,y_s+5))=\kappa_i$. 故在此步骤中,只有满足结合域 $bd_S(t)=\kappa_i$ 和 $bd_E(t)==$ 的 Tile 才能稳定地结合到配置上. 当 $i=6$ 时,即在位置 (x_s-6,y_s+6) 处,由于暴露在外面的结合域为 $bd_W(S_6(x_s-5,y_s+6))==$, $bd_N(S_6(x_s-6,y_s+5))=r$,只有结合域为 $bd_S(t)=r$ 和 $bd_E(t)==$ 的 *Success* Tile 才能稳定地结合到配置上,进而完成整个组装过程. 如图 4(h)所示.

记 $t_{ij}\in U_i$ 是能够在第 i 步中在位置 $(x_s-j,y_s-i)\in Q_j$ 处结合到配置 S_i 上并能稳定存在的 Tile, $\Delta=\Sigma_{d\in D}\,g(bd(t),bd_{d^{-1}}(A(d(x,y))))\geqslant\tau$ 是 t Tile 能够稳定存在于配置上的条件. 具体的自组装算法记为 $TileCompute(S_0,n)$.

算法 2. $TileCompute(S_0,n)$.

$S_n\leftarrow S_0$;
for $i\leftarrow 0$ to n
  $S_n\leftarrow PerStep(S_{i-1},i)$;
end for
if $S_n(x_s-n,y_s+n)=Success$
  return S_n;
else
  return failure;
end if
$PerStep(S_{i-1},i)$;
for $j\leftarrow 0$ to m
  $S_i(x_s-j,y_s+i)=t_{ij}$;
end for
return S_i.

2.6 计算结果

本文提出的完美匹配自组装模型经 2.5 节的 Step1 自组装产生结果序列的过程中,对每个位置 $(x,y)\in Q_1$, U_1 中可能会有多个 Tile 能够结合到该位置上. 而对于同一个 Tile t, $t\in U_1$,可能也能够结合到不同的位置 $(x,y)\in Q_1$ 上. 也就是说位置 $(x,y)\in Q_1$ 和 Tile $t\in U_1$ 是多对一的关系. 所以,在结果序列中可能会出现重复的顶点 Tile,而这样的结果序列显然不是完美匹配. 为了将完美匹配的结果序列与这些非完美匹配的结果序列分离开来,本文设计了结果序列检测部分. 若结果序列是完美匹配,最终将在组装体的左上角出现一个标记 Tile——*Success* Tile;若不是完美匹配,组装体将在下一步升温过程中溶解,如图 5(a)所示:

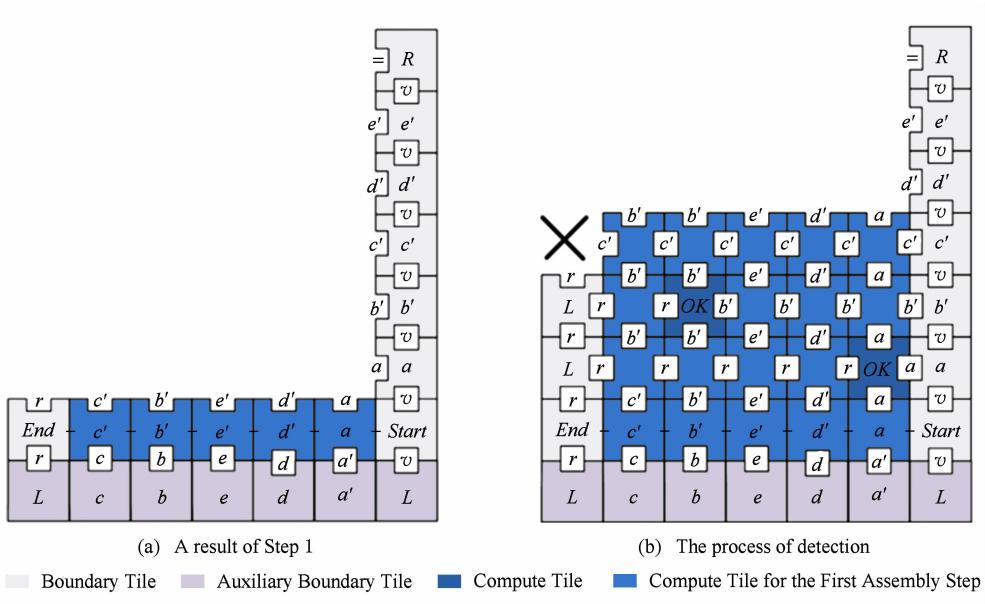


Fig. 5 Self-assembly model for perfect matching.

图 5 完美匹配自组装模型

图 5(a)是种子配置经 Step1 自组装后形成的结果序列. 由图 5(a)可知,在结果序列中 b' Tile 重复出现,而 c' Tile 没有出现,显然这个结果序列不是完美匹配. 图 5(b)是种子配置的检测部分对结果序列进行检测的自组装模型.

由图 5(b)可知,在对 c' Tile 进行检测时,由于没有在结果序列中扫描到自身,故 $bd_W(c')=c$ 将会一直向左传递. 在位置 (x_s-6,y_s+3) 上,暴露在 W 和 N 的结合域 $bd_W(S_2(x_s-5,y_s+3))=c'$, $bd_N(S_2(x_s-6,y_s+2))=r$,而 U_3 中没有结合域为 $bd_S(t)=r$ 和 $bd_E(t)=c'$ 的 Tile 能够结合到该位置上,从而终止自组装运算. 检测完成后,使 $\tau=3$. 由结合函数的

定义可知 $\forall\sigma\in\Sigma_{pm}\backslash\{null,=,v\}$,$g_{pm}(\sigma,\sigma)=1$. 根据定义 3 可知,在 $\tau=3$ 的条件下,Tile $S_2(x_s-5,y_s+3)$会从组装体上分离下来. 同样地,其他 Tile 最终也将分离下来,从而使整个组装体溶解掉.

3 算法分析

本文根据完美匹配概念及图中顶点间的邻接关系,编码完美匹配自组装系统用到的计算 Tile. 对于一个具有 n 个顶点的二部图来说,首先利用算法 1 根据图 1 中各顶点间的邻接关系,编码 DNA Tile 并形成种子系统. 编码后的 DNA Tile 通过 2.5 节中的组装过程进行自组装. 若组装成功,将在组装体左上角出现红色的 *Success* Tile;否则,组装体将在下一步升温过程中溶解.

定理 1. 完美匹配自组装系统 $S_{pm}=\langle T_{pm},g_{pm},\tau_{pm}\rangle$计算终止时返回的自组装结构是一个完美匹配.

证明. 对于任意一个二部图,其顶点数为 n. 给定 $S_{pm}=(T_{pm},S'_{pm},g_{pm},\tau_{pm})$,$g_{pm}$如 2.2 节和 2.4 节中的定义,$\tau_{pm}=2$,$S'_{pm}$是种子配置,根据 2.3 节,该系统所需的所有 Tile 集合为 $\Gamma_+=\{D_i\}_{i=0}^{13}$,其中:

$D_0=\langle X',r,X',r\rangle$,

$D_1=\langle Y',X',Y',X'\rangle$,

$D_2=\langle X',X',X',r\rangle$,

$D_3=\langle\delta,-,\kappa,-\rangle$,

$D_4=\langle v,null,v,-\rangle$,

$D_5=\langle r,-,r,null\rangle$,

$D_6=\langle v,null,null,null\rangle$,

$D_7=\langle r,null,null,null\rangle$,

$D_8=\langle null,=,r,null\rangle$,

$D_9=\langle null,null,v,=\rangle$,

$D_{10}=\langle r,r,r,null\rangle$,

$D_{11}=\langle v,null,v,Z'\rangle$,

$D_{12}=\langle\kappa,null,null,null\rangle$,

$D_{13}=\langle null,=,X',=\rangle$.

$S'_{pm}(x_s,y_s)=D_4$,$S'_{pm}(x_s-k,y_s-1)=D_7$,$S'_{pm}(x_s,y_s+k)=D_9$. $\forall i\in\{1,2,\cdots,k-1\}$,$S'_{pm}(x_s,y_s+i)=D_{11}$,$S'_{pm}(x_s-i,y_s-1)=D_{12}$. 那么种子配置 $S'_{pm}:\mathbb{Z}^2\to\Gamma_+$ 是:$S'_{pm}(x_s,y_s)=Start$,$S'_{pm}(-k,-1)=L$,$S'_{pm}(0,k)=R$,$S'_{pm}(0,i)=D_{11}$,$S'_{pm}(-i,-1)=D_7$.

对其他位置$(x,y)\in\mathbb{Z}^2$,$S'_{pm}(x,y)=empty$. 由 2.5 节可知,$S_0=S'_{pm}$经过 t 步自组装运算得到新配置 S_t.

由 2.4 节可知:1)种子系统中第 0 行的 D_{12} 与第 0 列的 D_{11}(或者 D_3 Tile)是二部图中的顶点 Tile,且两者交集为空;2)只有当 D_3 Tile 与 D_{12} Tile 所代表的顶点相邻,D_3 Tile 和 D_{12} Tile 才能在自组装运算中稳定结合. 由 1)和 2)和完美匹配的 3 个基本要求可知,若自组装计算 2.5 节的 Step1 完成后形成结果序列中没有重复的顶点 Tile,则这个自组装结果就是一个完美匹配. 在该系统中,除 Step1 之外,其余所有步骤都是在检测 Step1 产生的结果中没有重复序列.

接下来证明检测步骤中能够把不成功的计算结果检测出来而不被留在最终计算结果里.

对于任意含有重复 Tile 的 S_1,根据 2.4 节描述的检测过程中,$\forall i\in\{1,2,\cdots,k-1\}$,$i$ 表示 S_1 的横坐标,结合到位置$(-i,d-1)$上的 S_1 是 D_1 Tile. 当 $i=k$ 时,对于位置$(-k,d-1)$,只有 E 的结合域是 Z_{d-1}、S 的结合域为 r 的 Tile t 才能稳定地结合上去,但 Γ_+ 中没有满足条件的 Tile t. 若假设在位置$(-k+1,d-1)$处的 Tile 为 Tile t',那么根据 2.2 节中 g_{pm}的定义,此时 Tile t'的结合域强度和为 2(E 结合域强度为 1,S 结合域强度为 1,N 和 W 的结合域强度为 0). 在自组装运算过程中,系统温度 $\tau_{pm}=2$. Tile t'能够稳定结合在配置 S_d 上,但当系统温度升高到 $\tau_{pm}=3$ 时,根据定义 3 可知 Tile t'将会从 S_d 上分离下来. 证毕.

定理 2. 完美匹配自组装系统 $S_{pm}=\langle T_{pm},g_{pm},\tau_{pm}\rangle$的组装时间是 $O(n)$的.

证明. 本文提出的完美匹配自组织系统 Tile 系统$S_{pm}=\langle T_{pm},g_{pm},\tau_{pm}\rangle$,一个Tile集$\Gamma_+$,种子配置 $S'_{pm}:\mathbb{Z}^2\to\Gamma_+$. 当定义 1 中所有定义都满足时,那么 T_{pm}中的一个 Tile 能结合到 S'_{pm}上也能从 S'_{pm}上分离下来. 令 $V_0\in\mathbb{Z}^2$是 T_{pm}中的 Tile 能从 S_0 中分离下来的位置集合,S'_0 是所有在位置 V_0 处从 S'_{pm}上分离下来的 Tile 产生的配置. $W_0\in\mathbb{Z}^2$是 T_{pm}中 Tile 能结合到 S'_0 上所有的位置集合,对所有的 $w\in W_0$,U_w 是能在位置 w 处结合到 S'_0 上的 Tile 集合,$\hat{S}$ 是配置 S_1 的集合,S_1 满足以下条件:

$$\forall p\in S'_{pm},\ S_1(p)=S'_0(p),$$

$$\forall w\in W_0,S_1(w)\in U_w,$$

$$\forall p\notin S'_{pm}\cup W_0,S_1(p)=empty.$$

对所有的 $S_1\in\hat{S}$,我们就说 S_{pm}在 S'_{pm}上经过 1 步产生了 S_1,对于配置 $A_0,A_1,\cdots,A_{n/2+2}$来说,$\forall i\in\{1,2,\cdots,(n/2)+2\}$,都有 S_{pm}在 A_{i-1}上经过 1 步产生 A_i,那么就可以说 S_{pm}在 A_0 上经过$(n/2)+2$ 步产生了 $A_{(n/2)+2}$. 如果 S_{pm}在配置 A 上产生的唯一配置是 A 本身,那么 A 就被称作是一个最终配置;如果 A

是 S_{pm} 在 S'_{pm} 上产生的一个最终配置，那么 $(n/2)+2$ 就是 S_{pm} 在 S'_{pm} 上产生 A 的组装所需的步数，也就是说 S_{pm} 在 S'_{pm} 上产生 A 的组装时间是 $O(n)$. 证毕.

定理 3. 完美匹配自组装系统 $S_{pm}=\langle T_{pm}, g_{pm}, \tau_{pm}\rangle$ 所需的 Tile 个数是 $\Theta(n^2)$.

证明. 由 2.3 节和 2.4 节可知，完美匹配 DNA 计算自组装模型共需要 14 种 Tile 类型，如图 2 所示. 其中图 2(a)～(d)中的 4 种 Tile 类型以及图 2(e)和(f)中 2 种 Tile 类型，是所有完美匹配自组装系统都会用到的 6 个边界 Tile；图 2(g)～(j)分别是组装体的左、右、下和上的边界 Tile 类型，每种 Tile 类型需要 $n/2$ 个 Tile，共需 $2n$ 个 Tile；图 2(k)是将结果序列中的 Tile 信息向上传递的 Tile 类型，共需 $n/2$ 个 Tile；图 2(l)中 Tile 的作用是当 $bd_S(t)\neq bd_E(t)$ 时，将 S 和 E 的结合域分别向上和向左传递，故这种类型的 Tile 共需 $n/2\times n/2=n^2/4$ 个；图 2(m)中的 OK Tile 是当 $bd_S(t)=bd_E(t)$ 时，将 S 的结合域向上传递，故这种类型的 Tile 共需 $n/2$ 个；图 2(n)中所示的 Tile 类型所需的 Tile 个数与顶点个数及顶点度数有关. 假设一个有 n 个顶点的二部图，经过算法 1 的算法处理后，共有 k 个度数为 1 的顶点，$\forall j\in\{1,2,\cdots,k\}$，$\sigma_j$ 是与度数为 1 的顶点相邻接的顶点，σ_j 度数是 λ_j. $\forall i\in\{1,2,\cdots,n\}$，$\gamma_i$ 是顶点 β_i 的度数，那么图 2(n)中 Tile 类型的个数为 $\frac{1}{2}\sum_{i=1}^{n}\gamma_i-\frac{1}{2}\sum_{j=1}^{k}\lambda_j+k$，故整个自组装系统所用到的 Tile 的个数为 $\frac{1}{2}\sum_{i=1}^{n}\gamma_i-\frac{1}{2}\sum_{j=1}^{k}\lambda_j+k+n^2/4+3n+6$.

证毕.

4 结 论

本文针对二部图完美匹配问题提出了一种基于 DNA 计算自组装模型的算法. 根据二部图完美匹配的概念及二部图中顶点间的邻接关系，设计编码 DNA Tile 及种子系统，并利用 DNA 计算自组装模型本身所具有的高度并行性、海量存储能力、低能耗和组装的自治性等特性，使运算 DNA Tile 在一定控制条件下执行自治的组装运算.

本文以一个具有 10 个顶点的二部图(如图 1 所示)为例，介绍了该算法的步骤：1) 给出 DNA Tile 的编码设计方案，该方案对所有子图中的 1 度顶点 t 及与其相邻的顶点 t^{-1} 进行编码，使 t Tile 只能与 t^{-1} Tile 相结合. 由于这种处理方法避免了除 t 之外与 t^{-1} 相邻的其他顶点的 Tile 的设计，故降低了 DNA Tile 的个数. 同时，由于 t Tile 是 t^{-1} Tile 唯一能够结合的 DNA Tile，所以在 2.5 节的 Step1 之后，生成的结果序列中没有其他 DNA Tile 与 t^{-1} Tile 争夺与 t Tile 结合的机会，从而提高了结果序列的准确率. 另外，本文还设计了结果序列检测部分，通过对结果序列进行扫描，检测结果序列中是否存在重复的顶点 Tile，从而判断结果序列是否是完美匹配. 若不存在重复的顶点 Tile，则说明结果序列是完美匹配，那么最终将会在组装体的左上角出现红色的 *Success* Tile，结果在组装体的最上面一行读取；否则，结果序列不是完美匹配.

在本系统中，在编码过程中的每一步都需要从子图 $G(V,E)$ 中找出所有度数为 1 的顶点，并将 1 度顶点以及与 1 度顶点相邻的顶点和与它们相关联的边从图 G 中除去，并重置图 G.

利用 DNA 自组装进行计算是一种新的计算理念，通过设计具有不同功能的 DNA Tile，进而形成代表问题的解结构，从理论上展现了自组装模型应用于复杂问题的计算.

参 考 文 献

[1] Xu Jin, Bao Zheng. Neural networks and graph theory [J]. Scientia Sinica: Technologiea, 2001, 31(6): 533-555 (in Chinese)
(许进，保铮. 神经网络与图论[J]. 中国科学：技术科学，2001, 31(6): 533-555)

[2] Azadeh A, Faiz Z S, Asadzadeh S M, et al. An integrated artificial neural network-computer simulation for optimization of complex tandem queue systems [J]. Mathematics and Computers in Simulation, 2011, 82(4): 666-678

[3] Liu Yang. Database processing in quantum computers [D]. Beijing: Tsinghua University, 2008 (in Chinese)
(刘洋. 量子计算机中的数据库处理[D]. 北京：清华大学，2008)

[4] Colnaghi T D, Ariano G M, Facchini S, et al. Quantum computation with programmable connections between gates [J]. Physics Letters A, 2012, 376(45): 2940-2943

[5] Rahman M O, Hussain A, Scavino E, et al. DNA computer based algorithm for recyclable waste paper segregation [J]. Applied Soft Computing, 2015, 31: 223-240

[6] Sun Y H, Chen M Y. A DNA-based solution to the graph isomorphism problem using Adleman-Lipton model with stickers [J]. Applied Mathematics and Computation, 2008, 197(2): 672-686

[7] Ouyang Q, Kaplan P D, Liu S, et al. DNA solution of the maximal clique problem [J]. Science, 1997, 278(17): 446-449

[8] Xu J, Qiang X, Yang Y, et al. An unenumerative DNA computing model for vertex coloring problem [J]. IEEE Trans on Nanobioscience, 2011, 10(2): 94-98

[9] Beneson Y, Tamar P, Adar R, et al. Programmable and autonomous computing machine made of biomolecules [J]. Nature, 2001, 414(22): 430-434

[10] Shi X, Chen C, Li X, et al. Size-controllable DNA nanoribbons assembled from three types of reusable brick single-strand DNA tiles [J]. Soft Matter, 2015, 11(43): 8484-8492

[11] Shi X, Lu W, Wang Z, et al. Programmable DNA tile self-assembly using a hierarchical sub-tile strategy [J]. Nanotechnology, 2014, 25(7): 075602

[12] Feynman R P. There's plenty of room at the bottom [J]. Engineering and Science, 1960, 23(5): 22-36

[13] Adleman L. Molecular computation of solution to combinatorial problems [J]. Science, 1994, 266(5187): 1021-1024

[14] Winfree E. Algorithmic self-assembly of DNA [D]. Los Angeles: California Institute of Technology, 1998

[15] Zhang Xuncai. The research and applications of DNA computing by self-assembly [D]. Wuhan: Huazhong University of Science and Technology, 2009 (in Chinese)
(张勋才. 自组装 DNA 计算模型的研究及应用[D]. 武汉:华中科技大学, 2009)

[16] Chen Zhihua. Researches on several cryptological problems based on DNA computing by self-assembly [D]. Wuhan: Huazhong University of Science and Technology, 2009 (in Chinese)
(陈智华. 基于 DNA 计算自组装模型的若干密码问题研究[D]. 武汉:华中科技大学, 2009)

[17] Brun Y. Solving satisfiability in the tile assembly model with a constant-size tiletset [J]. Journal of Algorithms, 2008, 63(4): 151-166

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

[19] Tang Min, Guan Jian, Deng Guoqiang, et al. A new algorithm and application of solving maximum matching problem of bipartite graph [J]. Computer Systems Applications, 2012, 21(3): 72-75 (in Chinese)
(唐敏, 关键, 邓国强, 等. 一种求解二部图最大匹配问题新算法及其应用[J]. 计算机系统应用, 2012, 21(3): 72-75)

[20] Ying Guisheng, Cui Xiaohui, Dong Hongbin, et al. Quantum-cooperative method for maximum weight perfect matching problem of bipartite graph [J]. Journal of Computer Research and Development, 2014, 51(11): 2573-2584 (in Chinese)
(印桂生, 崔晓晖, 董红斌, 等. 量子协同的二分图最大权完美匹配求解方法[J]. 计算机研究与发展, 2014, 51(11): 2573-2584)

[21] Gao Lin, Ma Runnian, Xu Jin. The molecular algorithm of the matching problem based on plasmid DNA [J]. Progress in Biochemistry and Biophysics, 2002, 29(5): 820-823 (in Chinese)
(高琳, 马润年, 许进. 基于质粒 DNA 匹配问题的分子算法[J]. 生物化学与生物物理进展, 2002, 29(5): 820-823)

[22] Wang Shiying. DNA computing of bipartite graphs for maximum matching [J]. Journal of Mathematical Chemistry, 2002, 31(3): 271-279

[23] Dong Yafei. The study of some DNA computing sticker models [D]. Wuhan: Huazhong University of Science and Technology, 2003 (in Chinese)
(董亚非. 若干 DNA 计算粘贴模型的研究[D]. 武汉:华中科技大学, 2003)

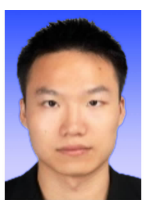
[24] Zhou Xu, Li Kenli, Yue Guangxue, et al. A volume molecular solution for maximum matching problem on DNA-based computing [J]. Journal of Computer Research and Development, 2011, 48(11): 2147-2154 (in Chinese)
(周旭, 李肯利, 乐光学, 等. 一种最大匹配问题 DNA 计算算法[J]. 计算机研究与发展, 2011, 48(11): 2147-2154)

[25] Buckley F, Lewinter M. A Friendly Introduction to Graph Theory [M]. Upper Saddle River, NJ: Prentice Hall, 2005: 42-43

[26] Lü Zongwei, Lin Zhenghui, Zhang Lei. Boolean mapping algorithm based on perfect matching of bipartite graph [J]. Journal of Computer-Aide Design & Computer Graphics, 2001, 13(11): 961-965 (in Chinese)
(吕宗伟, 林争辉, 张镭. 基于二部图完美匹配的布尔匹配算法[J]. 计算机辅助设计与图形学学报, 2001, 13(11): 961-965)

Lan Wenfei, born in 1966. Master and professor. Her main research interests include databases, software technology.

Xing Zhibao, born in 1987. Master candidate. Her main research interests include DNA computing.



Huang Jun, born in 1991. Master candidate. His main research interests include DNA computing.

Qiang Xiaoli, born in 1979. PhD and associate professor. Her main research interests include molecular computing and bioinformatics.