三元统计语言模型对基因表达载体设计的优化

方 刚[1],张社民[2]
FANG Gang[1], ZHANG Shemin[2]

1.西安文理学院 生物与环境工程学院,西安 710065
2.西安财经学院 信息学院,西安 710100
1.School of Biological and Environmental Engineering, Xi'an University, Xi'an 710065, China
2.School of Information, Xi'an University of Finance and Economics, Xi'an 710100, China

FANG Gang, ZHANG Shemin. 3-gram statistical language model optimization to expression vector design. Computer Engineering and Applications, 2016, 52(15):60-64.

Abstract: By assembling BioBrick according to a standard, complex genetic constructs composed of dozens of functional blocks can be made. But usually every category of BioBrick contains a few or many parts. With the increasing of BioBrick amount, the process of assembling more than a few of sets of BioBrick can be costly, time consuming and error prone. At the last step of assembling it is quite difficult to make decision which part should be selected. Based on statistical language model, a dynamic programming algorithm is carried out to solve the problem. The algorithm optimizes the results of a genetic design based on a grammatical model and figures out an optimal solution. In this way, redundant operations can be reduced and the time and cost required for conducting biological experiment can be minimized.
Key words: synthetic biology; statistical language model; dynamic programming; BioBrick

摘 要: 不断地将一些合成生物学标准"零件"以一定的标准装配,就可以得到由数十个功能片段组成的复杂表达载体。但是每一类的合成生物学标准"零件"数量众多,随着这些标准"零件"的不断开发,其数量也在进一步增加。在进行表达载体构建的最后阶段,从众多的"零件"中选择合适的以组装成功能性表达载体费时费力,并且容易发生错误。为解决这一问题,采用了自然语言处理的统计语言模型,并以该模型为基础应用动态规划算法优化表达载体设计,从众多的选项中找出最优者。利用这一方法可以减少进行生物学实验的冗余操作,从而减少表达载体构建过程中的花费。
关键词: 合成生物学;统计语言模型;动态规划算法;合成生物学标准"零件"
文献标志码: A **中图分类号:** TP39 doi:10.3778/j.issn.1002-8331.1604-0019

随着合成生物学的发展,数据量的大大增加,开发用于设计基因表达载体的工具软件变得十分必要。这一类工具不仅仅用于表达载体的设计,而且还可以用于基因及代谢网络研究[1-2]。比如GenoCAD[3],它是基于网络环境的而且免费使用,它通过编制一定的"语法",将每一个合成生物学标准"零件"看做一个词,用来设计所需的合成生物学构件[4]。在GenoCAD的设计中,启动子、核糖体结合位点、基因及终止子都属于各自的"词类",然后依据特殊编制的语法来设计分子生物学组件[5-6]。设计者们往往将已有的生物序列拆成"零件"来作为合成生物学的标准"零件"[7],而当设计者将这些属于不同词类的标准"零件"进行组装时,其过程通常耗时、费力而且容易发生错误。为了克服这些问题,研究者们引入了一些组装标准,属于同一组装标准的"零件"两侧序列包含相应的限制性酶切位点,使用同一组限制性内切酶和连接酶可以将同一标准的"零件"连接起来[8-9]。在一个设计软件中,也会模拟使用这些组装标准。对这些合成生物学标准"零件"的模拟组装可以在计算机中完全自动化[10],但是这样只是考虑了最基本的"语法",也忽略了以往的组装成功的案例。在进行表达载体设计时,

基金项目: 国家自然科学基金(No.61173113)。
作者简介: 方刚(1969—),通讯作者,男,理学博士,副教授,主要研究方向:生物信息学,E-mail:yuxiangqd@163.com;张社民(1962—),男,理学博士,教授,主要研究方向:图与组合优化。
收稿日期: 2016-04-06 **修回日期:** 2016-05-26 **文章编号:** 1002-8331(2016)15-0060-05

使用者可以根据自己需要编制“语法”，再根据一些具有生物学意义的设计规则变换设计结构，最后选择“零件”完成设计[11]。但是越来越多的“零件”被输入数据库，在设计的最后一步，设计者往往不知道选择哪个“零件”更为合适。为了解决这个问题，统计语言模型（Statistical Language Model，SLM）被引入设计中。统计语言模型（SLM）最初用于自然语言识别[12]，它用来估算一组词串成为一个正确语句的概率的大小。它最初也是目前最主要的应用是语音识别，除此之外还应用于机器翻译、分词、智能输入法及文本语音转换。本文介绍的统计语音模型通过统计BioBrick合成生物学标准件的一些参数，将BioBrick的组装过程转化为统计语音模型，然后使用动态规划算法找出合适的“零件”组装成最后的表达载体而完成设计。这一算法将iGEM（international Genetically Engineered Machine competition）竞赛设计的经验考虑进来，以减少时间和花费。这一方法不仅可以用来优化其他工具软件的设计方案，而且可以独立使用设计新的基因表达载体及分子生物学组件。

1　材料方法

使用链接http://parts.igem.org/das/parts/entry_points/下载BioBrick合成生物学组件信息，2014年1月的版本包含7 242个合成生物学组件。编写了一个 Perl脚本通过链接http://parts.igem.org/das/parts/features/?segment=part#分析并提取每个组件的信息。将每个组件的信息分解排列成特有的结构输入MySQL数据库。输入数据库后共分解提取出75 744个“零件”，这些“零件”包括基础“零件”（启动子、核糖体结合位点、基因编码序列、终止子及质粒序列）和复合“零件”。这些复合“零件”依然由基础“零件”组装而成。通过查询MySQL数据库提取出这些基础“零件”并统计它们的使用频数。同时编写一个Perl脚本和一些SQL语句分析复合“零件”，统计出相邻接的两个基础“零件”的使用频数。通过查询数据库，共提取出1 628个基础“零件”并符合RFC 23组装标准[13]。这意味着这些基础“零件”序列中（除了两侧的连接序列）不会包含这一组装标准使用的限制性酶切位点。这1 682个基础“零件”包含405个启动子，57个核糖体结合位点，57个终止子以及1 178个基因序列。这些基础“零件”将被用来设计基因表达载体。同时每一个基础“零件”的使用频率和每一对基础“零件”以及连续3个一组基础“零件”的使用频率都可以被计算出来。

1.1　语法模型

如何编制设计合成生物学组件的语法在以前的文献中已予以详细描述[14]，可以根据不同的目的编制不同的语法。但是基本的语法是一样的，即启动子-核糖体结合位点-基因-终止子（PRO（promoter）-RBS（ribosome-binding sites）-GEN（genes）-TER（terminator）），本文使用的语法与文献[14]使用的上下文无关的语法类似，但是增加了一些新的规则可以表达融合蛋白。

1.2　数学模型

在一些设计软件中，比如GenoCAD在设计的最后一步，设计者往往要从大量的基础“零件”中选择合适的“零件”以完成设计（图1），这一过程往往是困难的。为解决这一问题，本文引入了在语音识别、机器翻译、智能输入法中广泛应用的统计语言模型。

在这一模型中，一个句子（sentence，S）是否有意义并且合理基于其出现的概率。一个句子由一系列的词组成，在本文中一个“句子” S 就是一个由基础“零件”构成的生物学组件，这些基础“零件”就是组成“句子”的词，一个基础“零件” $part$ 就是一个词。因此，$S=part_1, part_2, \cdots, part_n$ 需要知道其发生概率 $P(S)$ 的大小。

$$P(S)=P(part_1, part_2, \cdots, part_n) \tag{1}$$

根据条件概率公式有：

$$P(part_1, part_2, \cdots, part_n)= P(part_1)\cdot P(part_2|\, part_1)\cdot P(part_3|\, part_1, part_2)\cdot \cdots \cdot P(part_n|\, part_1, part_2, \cdots, part_{n-1}) \tag{2}$$

式（2）中，$P(part_1)$ 指一个基础“零件”在一个设计中出现的概率。$P(part_2|\, part_1)$ 指 $part_1$ 出现在 $part_2$ 之前 $part_2$ 出现的概率。根据式（2），$part_n$ 出现的概率由所有出现在它之前的基础“零件”确定。其中 $P(part_1)$ 和 $P(part_2|\, part_1)$ 容易计算，但是计算 $P(part_3|\, part_1, part_2)$ 稍复杂，而计算 $P(part_n|\, part_1, part_2, \cdots, part_{n-1})$ 将非常困难，因为牵扯的变量太多导致条件过于复杂而难以估算。基于马尔科夫假设，可以认为一个“零件”在一个设计中出现的概率仅仅与它相邻的前一个“零件”相关。但是在表达载体设计时一个基因能否被有效地表达不仅与其RBS相关而且与其启动子密切相关。因此在真正的设计中认为一个“零件”在一个设计中出现的概率与它相邻的前两个“零件”相关。式（2）可以简化为：

$$P(S)=P(part_1)\cdot P(part_2|\, part_1)\cdot P(part_3|\, part_1, part_2)\cdot \cdots P(part_i|\, part_{i-2}, part_{i-1})\cdot \cdots P(part_n|\, part_{n-2}, part_{n-1}) \tag{3}$$

现在 $P(S)$ 即一个“句子”发生的概率就可以被计算出来了。式（3）就是统计语言模型中的三元模型（3-gram Model）。因此根据条件概率公式：

$$P(part_i|\, part_{i-2}, part_{i-1})=\frac{P(part_{i-2}, part_{i-1}, part_i)}{P(part_{i-2}, part_{i-1})} \tag{4}$$

将使用两个相邻基础“零件”出现的频率及连续三个基础“零件”出现的频率来估计上式中的 $P(part_{i-2}, part_{i-1},$

$part_i$) 和 $P(part_{i-2}, part_{i-1})$。其中

$$P(part_{i-2}, part_{i-1}, part_i) \approx \frac{Count(part_{i-2}, part_{i-1}, part_i)}{Count(all\_parts)}$$

$$P(part_{i-2}, part_{i-1}) \approx \frac{Count(part_{i-2}, part_{i-1})}{Count(all\_parts)}$$

可得

$$P(part_i | part_{i-2}, part_{i-1}) \approx \frac{Count(part_{i-2}, part_{i-1}, part_i)}{Count(part_{i-2}, part_{i-1})} \quad (5)$$

根据式(5),式(3)中任何一个成份都可以被计算出来。

设计的最后一步(图1),可以有很多个基础“零件”组合而成最后的设计。但是哪一种组合最合理而最有意义？根据统计语言模型理论,概率最大那一个将是最合理和最有意义的。在如图1所示网格结构中,可以有很多个候选路径产生“句子”,一条路径产生一个“句子”,一个“句子”就是一个设计 ($a\ path = a\ S = part_1, part_2, \cdots, part_n$)。最优的路径由 $PATH$ 表示：

$$PATH = \arg\max_{all\_S}(P(S))$$

为避免计算时内存溢出,对 $P(S)$ 取对数值：

$$PATH = \arg\max_{all\_S}(\ln P(S)) = \arg\max_{all\_S}(\ln(P(part_1) \times P(part_2 | part_1) \times \prod_{i=3}^{n} P(part_i | part_{i-2}, part_{i-1}))) =$$

$$\arg\max_{all\_S}(\ln P(part_1) + \ln P(part_2 | part_1) + \sum_{i=3}^{n} \ln P(part_i | part_{i-2}, part_{i-1})) \quad (6)$$

根据式(5),得到了下式：

$$P(part_i | part_{i-2}, part_{i-1}) = \frac{Count(part_{i-2}, part_{i-1}, part_i)}{Count(part_{i-2}, part_{i-1})}$$

$$P(part_2 | part_1) = \frac{Count(part_1, part_2)}{Count(part_1)}$$

$$P(part_1) = \frac{Count(part_1)}{Count(all\_parts)}$$

因为从一个相对稀疏的语料库中提取参数,零概率问题不可避免。为克服这一问题采用加1(拉普拉斯)数据平滑技术[15]。因此上式又变为：

$$P(part_i | part_{i-1}) = \frac{Count(part_{i-1}, part_i) + 1}{Count(part_{i-1}) + N} \quad (7)$$

$$P(part_i | part_{i-2}, part_{i-1}) = \frac{Count(part_{i-2}, part_{i-1}, part_i) + 1}{Count(part_{i-2}, part_{i-1}) + W} \quad (8)$$

式(7)中 N 表示二元组的个数(两个相邻基础“零件”出现的个数),式(8)中 W 表示三元组的个数(连续三个相邻基础“零件”出现的个数),式(7)和(8)将被用来计算式(6)中各个成分的值从而得出最优路径。最优路径 $PATH$ 将是在所有路径中具有最大概率的那一个。这里使用动态规划算法找出最优路径 $PATH$ 。

1.3　算法

在如图1的网格结构中找出最优的路径,这一路径将由一系列基础“零件”构成而它具有最大的出现概率。找最优路径的过程就是对式(6)求解,这里的动态规划算法源于维特比算法[16]。由三个步骤组成：

第一步,建立候选网格。

每一类“零件”对应一列,而每列中的每一个节点对应一个“零件”。在网格的开始和结束添加BEG和END列。在这两列中两个虚拟节点 B 和 E 被添加进来(图2),每一个节点是一个三元组 $<name, V, P>$,三元组的第一元素 $name$ 是基础“零件”的序列号,这一序列号对任何一个“零件”来说是唯一的。

第二步,填充网格。

从左至右填充网格,对于每一个三元组 $<name, V, P>$,V 和 P 被计算出来并填充相应值。V 由连续三列的所有节点进行组合运算的最大值填充,P 将存储与当前节点组合运算产生最大值的前一个节点的地址信息。

(1)对第一列,节点 B 使 $V=0$ 且 $P=$NULL 。

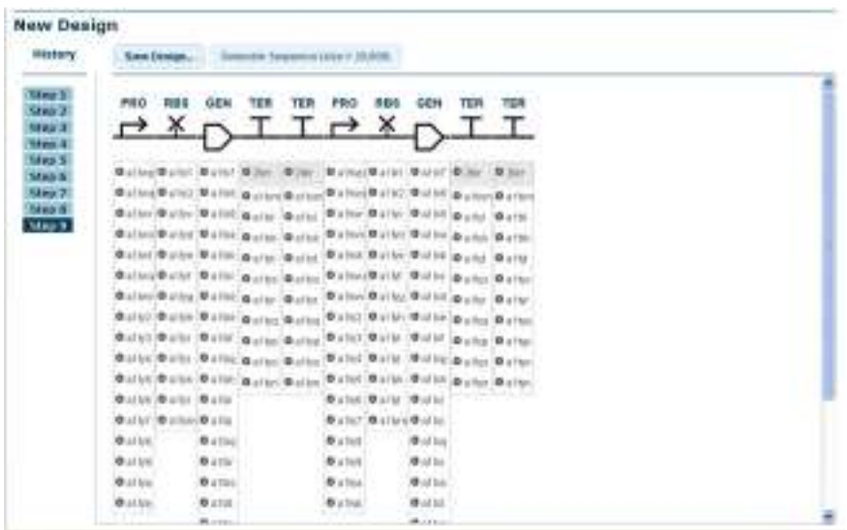

图1　GenoCAD设计的最后一步

(2)对第二列每一节点三元组 $<name, V, P>$ ($name \in$ {I0500, R0011, …, R0040, …}) 将与 B 节点组合计算其 V 值和 P 。

$$V = V_B + \ln P(part) = \ln P(part)$$

$$P = address\_of\_B$$

(3)对第三列每一节点三元组 $<name, V, P>$ ($name \in$ {R0032, R0034, …, R0041, …}) 将与第二列中的所有节点组合并计算其 V 值和 P 。

$$V = \max\{\ln P(part \mid part_{\text{prior}})\}$$

$P = address\_where\_V\_comes\_from$ (当前的 P 存储与当前节点组合取最大值前一个节点的地址)

(4)对第四列每一节点三元组 $<name, V, P>$ 将与前两列中的所有节点组合并计算其 V 值和 P 。

$$V_{\text{Current}} = \max\{V_{\text{Before}} + V_{\text{Previous}} + \ln P(part \mid part_{\text{Before}}, part_{\text{Previous}})\}$$

$P_{\text{Current}} = Address\_of\_V_{\text{Previous}}\_belong$ (当前的 P 存储与当前节点组合取最大值前一个节点的地址)

$P_{\text{Previous}} = Address\_of\_V_{\text{Before}}\_belong$ (前一个节点的 P 存储与当前节点组合取最大值第一个节点的地址)

(5)重复(4)，在当前列中的每一个节点都与前两列的每一个节点组合计算其 V 值和 P 。

(6)在END列，E 节点的 V 将由选自前一列的最大值填充，P 将存储前一列那个最大值节点的地址信息。

第三步，回溯找出最优路径 $PATH$ 。

从节点 E 开始不断找出前面节点的 P (图2)，最终最优路径 $PATH$ 将是具有最大概率的那一个，其产生的“句子” S 就是最后设计的具有生物学意义的分子生物学组件。如果 S 的长度是 L 而一列中节点的个数最多是 D ，该算法的算法复杂度是 $O(L \cdot D^3)$ ，对于二元模型算法复杂度是 $O(L \cdot D^2)$ ，但是理论上讲二元模型的准确率要差一些。穷举算法的算法复杂度是 $O(D^L)$ 。

2　结果

为展示如何将BioBrick基础“零件”组装成功能性的合成生物学组件，挑选了一个可以产生香蕉气味的质粒(http://parts.igem.org/Part:BBa_J45900)。该质粒由麻省理工学院参加2006年iGEM竞赛的参赛队设计并实施。该组件包含两个表达盒：一个盒子包含BAT2和THI3基因，另一个表达盒的产物催化前面基因的产物而使大肠杆菌发出香蕉的气味。该装配算法由一个Perl脚本执行，首先编制相应的“语法”，然后根据“语法”设计合成生物学组件，具体语法见表1。

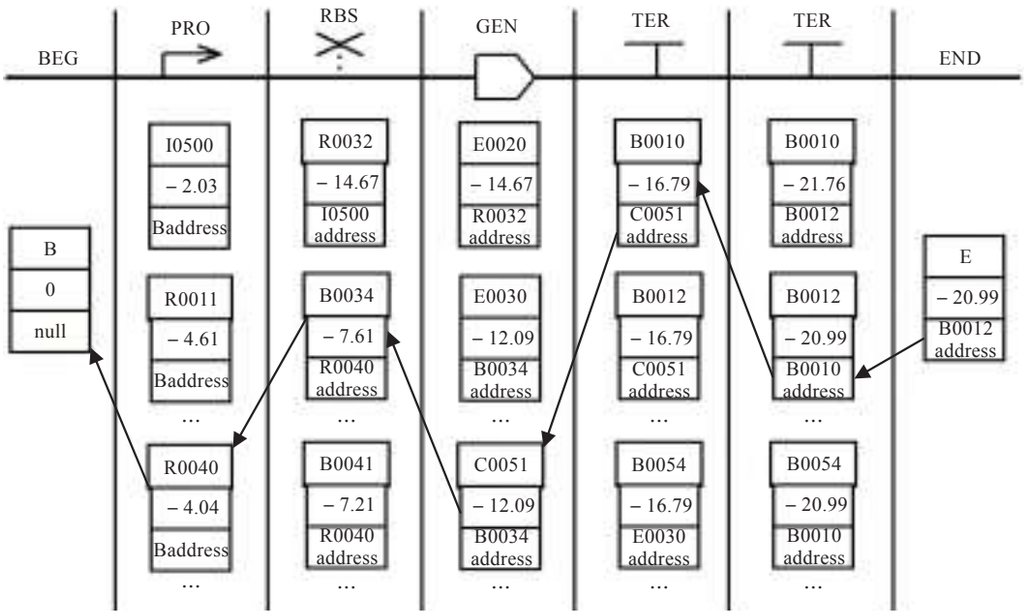

图2　建立网格，填充网格及回溯过程

表1　用于设计表达融合蛋白的“语法”

Rule	Comments	Left term	Right term
1	Transform a cassette (Cass) into two cassettes (Cass)	Cass	Cass-Cass
2	Reverse the sequence orientation of a cassette (Cass)	Cass	[Cass]
3	Transform a cassette (Cass) into a promoter (PRO), a cistron (Cis), and a terminator (TER)	Cass	PRO-Cis-TER
4	Transform a cassette (Cass) into promoter and cistrons (PRO-Cis)	Cass	PRO-Cis
5	Transform a cistron (Cis) into a rbs (RBS) and a gene (GEN)	Cis	RBS-GEN
6	Transform a terminator (TER) into two terminators (TER)	TER	TER-TER
7	Transform a cistron (Cis) into two cistron (Cis)	Cis	Cis-Cis

其中规则1表示将一个表达盒变成两个,规则2表示翻转一个表达盒,规则3表示将一个表达盒变成一个启动子加一个阅读框加一个终止子,规则4表示将一个表达盒变成一个启动子加一个阅读框,规则5表示将一个阅读框变成一个核糖体结合位点与一个基因,规则6表示将一个终止子变成两个,规则7表示将一个基因变成两个。设计的具体过程是这样的:首先需要一个表达盒Cass,依据规则1它可以变成两个表达盒Cass-Cass。对第一个、第二个表达盒使用规则4和3,该设计变成PRO-Cis-PRO-Cis-TER。对第一个表达框Cis使用规则7,对第二个Cis使用规则5,该设计变为PRO-Cis-Cis-PRO-RBS-GEN-TER。最后一步对两个Cis使用规则5,对TER使用规则6,最后的设计就是PRO-RBS-GEN-RBS-GEN-PRO-RBS-GEN-TER-TER。在输入该设计之前,先确定需要表达的基因,然后由算法选择合适的"零件"完成设计,输入的设计是:

PRO.RBS.GEN(J45008).RBS.GEN(J45009).PRO.RBS.GEN(J45014).TER.TER

该装配算法由一个Perl脚本执行,它给出序列:

R0040-B0030-J45008-B0030-J45009-R0040-B0030-J45014-B0010-B0012

第二次执行算法,当在第一列排除启动子R0040,算法给出序列:

R0011-B0030-J45008-B0030-J45009-R0040-B0030-J45014-B0010-B0012

由这些基础"零件"构成的序列就是产生香蕉气味这一合成生物学组件的真实组成。可见三元模型只需两步就可以给出真实组成。如果进行其他的设计并执行算法,该方法将给出一个优化的结果,这一结果采用了以往设计的经验。如果需要更多的选项,可以排除一些"零件"并重新执行算法,它将给出其他一些优化的结果供选择。如果已知某些"零件"是确定相邻接的,使用者可以先确定这些连接然后执行算法。

3 讨论

在本文中,统计语音模型被用于合成生物学基因片段组装而产生表达载体,通过将BioBrick基础"零件"的装配过程转化成一个三元统计语言模型,然后执行动态规划算法找出最优结果。算法可被迭代从而给出不同的优化结果供选择。这一方法不仅可以用来优化其他合成生物学软件设计结果,而且可以独立使用来模拟装配合成生物学基因片段产生表达载体。依据一定的语法输入不同类的合成生物学基础"零件",算法依据以往经验自动选择合适的"零件"装配成合成生物学组件。采用这一方法可以减少真实装配过程的冗余操作,从而节省时间和费用。如前所述,该方法采用三元统计语言模型,这意味着每一个"零件"只与它前面两个相邻的"零件"有联系。然而在真实的分子生物学环境中,一个基因能否高效表达不仅与其核糖体结合位点、启动子有关,而且与其他调控序列有关。考虑 N 元模型,这意味着每一个"零件"与它前面 $N-1$ 个"零件"有关系,但是这时条件概率是非常难以计算的。当 $N=4$ 或5时,尽管在其他统计语音模型应用范例中(如机器翻译、分词、智能输入法)准确率会大大提高,但是计算量也大大增加,这时需要功能强大的计算机[12]。下一步,将开发一个4元模型并且将质粒序列考虑进来,从而更高效地模拟合成生物学基因片段的组装过程并使真实的组装变得更为方便快捷。

当计算条件概率时,本文使用了拉普拉斯数据平滑技术解决零概率问题。但是在其他统计语言模型的应用中,这一技术效果不佳[15]。本文使用这一数据平滑技术主要考虑在合成生物学中某些"零件"极少被连接在一个表达质粒中。因为目前统计语言模型在合成生物学中鲜有应用,无从知道哪一种数据平滑技术更有效。但是拉普拉斯平滑的缺点是明显的,比如对未出现事件给予过多概率空间,预测二元模型概率较差[15]。下一步开发四元模型时,将扩大语料库,并考虑Good-Turing估计、卡茨退避法、线性插值法[17-18]等数据平滑技术以提高准确率。正如前述,从iGEM(international Genetically Engineered Machine competition)网站下载了一个相对稀疏的语料库,目前考虑将语料库扩展到常用的、商业化的表达载体上统计相应的参数,这样统计语言模型可以在合成生物学中更广泛地应用并得到检验。但是对这些合成生物学片段进行描述的术语还没有完全统一,因此发展合成生物学开放语言(Synthetic Biology Open Language,SBOL)同时开发与之相应的数据挖掘技术变得十分必要。

参考文献:

[1] Goler J A,Bramlett B W,Peccoud J.Genetic design:rising above the sequence[J].Trends Biotechnol,2008,26:538-544.

[2] Graslund S,Nordlund P,Weigelt J,et al.Protein production and purification[J].Nat Methods,2008,5:135-146.

[3] Czar M J,Cai Y,Peccoud J.Writing DNA with GenoCAD[J].Nucleic Acids Res,2009,37.

[4] Cai Y,Wilson M L,Peccoud J.GenoCAD for iGEM:a grammatical approach to the design of standard-compliant constructs[J].Nucleic Acids Res,2010,38:2637-2644.

[5] Isaacs F J,Dwyer D J,Ding C,et al.Engineered riboregulators enable posttranscriptional control of gene expression[J].Nat Biotechnol,2004,22:841-847.

[6] Gardner T S,Cantor C R,Collins J J.Construction of a genetic toggle switch in Escherichia coli[J].Nature,2000,403:339-342.

(下转140页)