第29卷 第17期 计 算 机 工 程 2003年10月
Vol.29 №17 Computer Engineering October 2003
·人工智能及识别技术· 文章编号：1000—3428(2003)17—0122—02 文献标识码：A 中图分类号：TP301.6

基于隐式马尔可夫链的基因发现模型和算法

章 铭，陆菊康
（上海大学计算机学院，上海 200072）

摘 要： 探讨了隐式马尔可夫链在基因发现中的应用。提出了一个基于GHMM（泛化的隐式马尔可夫链）的基因发现系统的简化的模型，论述了用该模型和扩展的Viterbi算法发现基因的方法，介绍了用于描述编码区和非编码区及信号的模型和实现。

关键词： 基因发现；HMM；GHMM

HMM-based Gene-finding Model and Algorithm

ZHANG Ming, LU Jukang
(College of Computer, Shanghai University, Shanghai 200072)

【Abstract】 This article describes the application of Hidden Markov Chain in the gene-finding system. A simplified model of gene-finding system, which is based on GHMM, is presented. Then it describes the method of finding gene with this model and modified Viterbi algorithm. Also it introduces the models for coding and non-coding regions and signals.

【Key words】 Gene finding; HMM; GHMM

随着人类基因组工程的完成以及越来越多的生物体的DNA被测序出来，从大量的序列中获取有价值的信息变得日益重要。而其中最有意义的是识别出蛋白质的编码区域，并由此推测出完整的基因结构和相应蛋白质的结构。在过去十几年的时间里，研究人员已开发出了许多基因发现(识别）系统，其中的一些得到了广泛的应用，如GeneMark[4]、GenScan[3]、Genie[2]等。这些系统采用了各种各样的技术，包括隐式马尔可夫模型、神经网络、动态编程和决策树等。其中基于隐式马尔可夫模型（HMM）的算法得到了广泛的研究和应用。本文论述基于HMM的扩展形式、GHMM的基因发现系统的框架和实现中的若干问题。

1 模型

1.1 HMM和GHMM基础

HMM是马尔可夫链的推广，和马尔可夫链一样可看作是一状态机，由一组状态及相关的转移组成。其与普通的马尔可夫链的不同之处在于，它的每一个状态也是一个随机过程，用以描述状态和输出的观察值之间的统计对应关系。因此，马尔可夫链输出的是一状态序列，而HMM输出的则是一观察值序列，产生这一观察值序列的路径（状态序列）是不可见的（隐含的），因此称其为“隐式的”。

HMM可表示为$\lambda = \{A, B, \pi\}$。其中π表示初始状态概率分布矢量。A代表状态转移概率分布矩阵$\{a_{ij}\}$ ($1 \leqslant i, j \leqslant N$,N是模型中状态的数目)，$a_{ij}=P[q_{t+1}=S_j \mid q_t=S_i]$表示从状态$S_i$转移到$S_j$概率。B代表观察值概率矩阵$\{b_j(k)\}$ ($1 \leqslant j \leqslant N$，$1 \leqslant k \leqslant M$，M是观察值的数目，即离散字母表的大小)，$b_j(k) = P[v_k \text{ at } t \mid q_t=S_j]$表示在时刻t，状态$S_j$生成观察值$v_k$的概率。

对于一观察值序列$O=o_1 o_2 \ldots o_T$，假设有状态序列$Q=q_1q_2 \ldots q_T$，则在模型λ中，O由Q生成的概率是

$$P(O \mid Q, \lambda) = \prod_{t=1}^{T} P(o_t \mid q_t, \lambda) = b_{q1}(o_1) * b_{q2}(o_2) * \ldots * b_{qT}(o_T)$$

而状态序列Q的概率是

$$P(Q \mid \lambda) = \pi_{q1} * a_{q1q2} * a_{q2q3} * \ldots * a_{qT-1qT}$$

由于$P(O \mid \lambda) = P(O \mid Q, \lambda) P(Q \mid \lambda)$，因此在给定模型$\lambda$的条件下，生成观察值序列O的概率就是

$$P(O \mid \lambda) = \sum_{\text{all } Q} P(O \mid Q, \lambda) P(Q \mid \lambda)$$
$$= \sum_{q1,q2,\ldots,qT} \pi_{q1} * b_{q1}(o_1) * a_{q1q2} * b_{q2}(o_2) * \ldots * a_{qT-1qT} * b_{qT}(o_T)$$

通常对于某一观察值序列O，有多条路径可生成它。而生成O的最优的（也就是能最好地“解释”O的）状态序列Q往往揭示了模型隐含的部分，解释了被模型所描述的事物的结构和特性。一种广泛使用的确定最优路径的办法是选择单个最佳状态序列（single best state sequence），也就是使$P(Q \mid O, \lambda)$达到最大值的状态序列。这可由Viterbi算法计算得到。在基因发现领域，观察值为4种脱氧核苷酸（A，T，C，G），而HMM模型的状态则对应于基因的各个功能序列段。而基因发现的方法就是对于一条DNA序列，通过Viterbi算法计算出其最优的状态序列（路径）。最终得到的路径就描述了该DNA序列中的预测的基因结构。关于HMM更详细的论述见参考文献[1]。

GHMM（Generalized HMM）是HMM的扩展形式。标准的HMM的每个状态产生一个观察值，而GHMM的每个状态则可产生0或多个观察值。这使得GHMM能更直观地描述基因的结构，并使框架有更好的可扩展性。

1.2 用于基因发现的GHMM框架

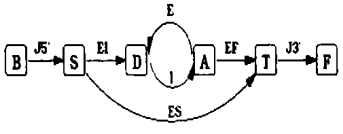

图1 多外显子基因的GHMM框架

图1是用于多外显子基因的GHMM框架。与一般的状态机不同，这里代表了状态机中的状态，节点代表了状态之间的转移，那么在一个分析中，将各边（状态）连接起来就可得到一条完整的基因序列。在GHMM中，状态和转移都可具体地对应到基因的各个组成部分。其状态的具体含义是：

作者简介： 章 铭（1977—），男，硕士生，研究方向：生物信息学，数据仓库；陆菊康，副教授

收稿日期： 2002-09-23

J5'－5'端不翻译区，J3'－3'端不翻译区，EI－初始外显子，E－内部的外显子，I－内含子，EF－末尾的外显子，ES －单外显子。节点对应于基因中的信号：D－5拼接点，A－3'拼接点，S－翻译开始区，T－翻译结束区。而B、F则是整个状态机的起始和终止位点。

当给定了一条候选的DNA序列X，找到一路径（称为分析，parse）Φ，使该分析Φ相对于其他分析来说，生成X的概率最大。该Φ中的各个状态即对应于基因的各个功能子序列，而Φ就描述了预测的基因的结构。可将这些概念形式化如下：

设M为一GHMM模型。由于GHMM的每一状态可生成多个符号（观察值），因此，定义X为一有序的子序列集合$\{x_1,x_2,\cdots,x_k\}$，使得X是各子序列的连接：

$X=x_1x_2\cdots x_k$

并定义分析为一有序的状态/序列对的集合：

$\Phi=\{\{q_1,x_1\},\{q_2,x_2\},\cdots,\{q_k,x_k\}\}$

其中q_i（$1\leqslant i\leqslant k$）是分析Φ的第i个状态(即模型中的边)，对应于基因的功能单元。而x_i就是X中的第i个子序列，是与该功能单元q_i相对应的子序列。这里规定q_1是离开开始节点B的边，q_k是指向结束节点F的边。可计算p[X，Φ]如下（隐含的以M为条件）：

$$p[X,\Phi]=P(q_1|B)(\prod_{i=1}^{k}P(x_i|q_i))(\prod_{i=1}^{k-1}P(q_{i+1}|node(q_i)))$$

$node(q_i)$是边q_i指向的节点。每个$P(x_i|q_i)$项都称为“内容检测器”（content sensor），是按q_i的模型生成x_i的概率。每个状态自身可以是任意的模型，如HMM、GHMM或神经网络。GHMM将系统层的模型与每个状态的模型（“内容探测器”）的实现细节相分离，使系统模块化，提高了整个系统的可扩展性，也使得能够用任意合适的模型来描述基因各个功能单元。分析序列X，就是要找到一个Φ，使得P[X,Φ|M]最大。这可用Viterbi 算法来实现。

2 系统设计和实现

2.1 状态的模型

在GHMM中，状态对应于基因的功能子序列，如内含子，外显子和5'、3'端的不翻译区。在基因发现领域，用于计算这些序列的模型称为“内容检测器”。这些模型本身也可以是HMM、GHMM。这里以编码区(包括EI、EF、E、ES)和非编码区(包括I、5'3'UTR)为例，介绍相应状态所采用的模型。

编码区就是基因中编码蛋白质的部分，每3个核苷酸（密码子、codon）编码一个氨基酸。而非编码区则是基因中不编码蛋白质的部分。在进化选择的作用下，各个子序列区域就有了不同的核苷酸使用特性。就可以利用不同类型的马尔可夫随机过程（或其他的模型，如神经网络）来描述这些子序列。

非编码区域一般使用同构马尔可夫模型（homogeneous Markov model）。一阶的同构马尔可夫模型包括：(1) 初始概率向量PN_0；(2) 转移矩阵PN，这些参数可由训练集合中非编码区的单核苷酸和二核苷酸的数量来计算。

由于在编码区中每3个核苷酸(密码子)编码一个氨基酸，随机过程就需要捕捉密码字每个位置的信息，因此编码区使用异构马尔可夫模型(inhomogeneous Markov model)。一阶的异构马尔可夫模型包括：(1) 3个初始状态概率向量：PI_0，$P2_0$，$P3_0$，各自对应密码子中的一个位置。每个向量包括元素PI_{0i}，$P2_{0i}$，$P3_{0i}$，i＝1，2，3，4，分别代表了4种核苷酸；(2) 3个转移矩阵P1、P2、P3,包括元素PI_{ij}、$P2_{ij}$、$P3_{ij}$，i，j＝1，2，3，4。

当给定一段DNA序列$S=s_1s_2\cdots s_n$，s_i是核苷酸，并设n是3的倍数。则S在非编码区的概率是

$$P[S|NON]=PN_0(s_1)*PN(s_2|s_1)*\cdots*PN(s_n|s_{n-1})$$

S在编码区的概率根据S的第一个核苷酸在密码子中的位置（1，2，3），可分成3种互斥的结果。

$$P[S|COD1]=P1_0(s_1)*P1(s_2|s_1)*P2(s_3|s_2)*P3(s_4|s_3)*\cdots*P2(s_n|s_{n-1})$$
$$P[S|COD2]=P2_0(s_1)*P2(s_2|s_1)*P3(s_3|s_2)*P1(s_4|s_3)*\cdots*P3(s_n|s_{n-1})$$
$$P[S|COD3]=P3_0(s_1)*P3(s_2|s_1)*P1(s_3|s_2)*P2(s_4|s_3)*\cdots*P1(s_n|s_{n-1})$$

由于密码子是由3个核苷酸组成的，因此二阶的马尔可夫模型是描述编码区的模型的最小阶数。但更高阶的模型需要更多的训练数据。我们采用了五阶的马尔可夫模型，它可以捕捉到两个相邻密码字的序列信息。通过对上述一阶的异构马尔可夫模型进行扩充，即可获得五阶马尔可夫模型。

2.2 信号的检测

信号是基因中的功能位点，如剪切位点、启动子。它通常是由几个和几十个核苷酸组成的序列片断（或由多个这样的片断组成），是基因不同功能单元序列之间的分界，标志着模型中状态之间的转移的发生，如剪切位点是内含子和外显子的分界，而启动子则标志了整个基因的起始。因此，在基因发现的过程中，正确的检测信号是非常重要的。

和检测功能子序列的“内容检测器”相对应，用以检测信号的模型称为“信号检测器”(signal sensor)。它既可以作为“内容检测器”的一部分，也可以通过修改公式，将其作为独立的项并入公式中，如Genie。

用于剪切位点的“信号检测器”通常采用权值矩阵方法（WMM，Weight Matrix Method）。在这种方法中，长度为λ 的信号中的核苷酸被认为是由与位置相关的概率分布独立生成的。当给定了一个信号模型“＋”和一段序列$X=x_1\cdots x_\lambda$，则这段序列的概率就是

$$P[X|+]=P'_{WMM}(X)=\prod_{i=1}^{\lambda}P^{(i)}{}_{x_i}$$

其中$P^{(i)}{}_j$是在信号位置i上生成核苷酸j的概率。我们采用了对WMM的扩展，权值阵列模型（WAM，Weight Array Model）。它实际上是一个一阶异构马尔可夫链，其中位置i的概率分布依赖于位置i－1上的核苷酸种类。

2.3 用Viterbi 算法寻找基因

在得到各个子模块之后，就可利用Viterbi 算法在给定的DNA序列中发现基因了。Viterbi 算法[1]是HMM中用以计算“最佳”路径的动态编程算法（Dynamic Programming）。由于这里用到的GHMM模型是对标准HMM模型扩展，因此对算法的实现过程做了相应的修改。具体的算法如下：

首先给所有的状态 (J5', EI, E, I, EF, J3') 编号为q_1，q_2、q_3、q_4、q_5、q_6。子序列$s_{i,j}$在状态q_k下的概率是$P(s_{i,j}|q_k)$。并设$\gamma_i(j)$为在位置j结束于状态Qi的子序列$S_{1,j}$的最佳分析的联合概率。并设给定序列的长度为L。

(1) 初始化：由于只寻找完整的基因,即以状态J5' (q_1)开头的分析。因此有

$$\gamma_i(1)=P(q_i|B)P(s_{1,1}|q_i),\ \gamma_i(1)=0\quad(2\leqslant i\leqslant 6)$$

(2) 递归：

$$\gamma_i(j)=\max_{1\leqslant k\leqslant i}[\gamma_i(k)\,P(q_i|node(q_i))P(s_{k+1,j}|q_i)]$$
$$(1\leqslant i\leqslant 6,2\leqslant j\leqslant N,2\leqslant s\leqslant 6)$$

（下转第145页）

运行在这部分CPU上的操作系统有一些不同于普通操作系统的要求，它必须考虑：

1) 支持多个可执行环境。每个可执行环境能对外提供一定的应用程序接口(APIs)，不同的应用程序接口能提供不同的应用与服务。同时，应用程序接口的版本能方便地更新、修补Bug、增加新功能、淘汰旧功能。这种设计也有利于提高网络安全性；

2) 程序要求访问不同的系统资源，如CPU资源、状态存储等，调度算法要能满足许多不同的资源特性；

3) 支持不同资源用户之间的合作和竞争，支持资源隔离和共享，提高安全性和充分利用计算资源。

主动网络最初的研究一般选用通用的操作系统，如Linux或Solaris，所有功能都在一个处理器和一个统一的操作系统下完成。区分网络功能与普通的计算功能可简化操作系统设计的难度，具体效果还有待于进一步的研究与比较。

3 相关研究

麻省理工的ANTS[2] 研究比较早，用数据舱方式实现，流动代码使用Java字节代码，在Java虚拟机上解释执行，缺点是效率比较低。后来的研究PAN[6] 采用了机器代码与代码的缓存，以改善运行效率。宾夕法尼亚大学的研究[3] 在安全性方面做了比较多的工作，有安全的自举结构AEGIS[5]，其PLANet[3]也是基于数据舱方式的。哥伦比亚大学的NetScript主动网络系统是基于可编程路由器方式的，主要考虑其数据流的处理和建立在TCP端口上的远程控制协议。

Bowman对数据包的处理采用了分类算法、支持针对流的多线程处理，采用了实时处理器调度以取得决定性的性能。Janos是一个用于主动网络节点的操作系统，它的主要关注是强大的资源管理和对用Java编写的不信任主动应用的控制。文献[5]在主动网络中引入标签，用类似MPLS的技术改善转发性能；文献[4]研究了用可编程硬件来达到高性能和可扩展性。

4 结论

本文讨论了实现主动网络需要考虑的安全性、性能、互通性和操作系统的问题，提出了一些设计思路。文中参考了大量主动网络的最新的研究，也借鉴了传统路由器、多服务接入平台的设计经验和教训。

与传统网络一样，主动节点也有低端和高端之分，两种类型应该在设计上有所不一样。另外，对具体的服务模式而言，要求在每一个主动节点上都进行计算的情况很少，这样计算任务在网络节点上的分配也是需要考虑的问题。这些问题与本文讨论的安全性、性能、互操作和操作系统等问题，都有待于进一步在研究中比较和优化。

参考文献

1 Tennenhouse D L, Smith J M, Sincoskie W D, et al. A Survey of Active Network Research. In IEEE Communications Magazine, 1997,35 (1): 80-86

2 Wetherall D J, Guttag J V, Tennenhouse D L. ANTS: A Toolkit for Building and Dynamically Deploying Network Protocols. Presented at the IEEE Open Architectures and Network Programming, 1998: 117

3 Hicks M, Moore j T, Alexander D S, et al. PLANet: An Active Internetwork. In Proc. INFOCOM' 99 Eighteenth Annual Joint Conference of the IEEE Computer and Communications Societies, 1999,3: 1124

4 Wolf T, Turner j S. Design Issues for High-performance Active Routers. In IEEE Journal on Selected Areas in Communications, 2001,19 (3): 404 -409

5 Wolf T, Decasper D, Tschudin C. Tags for High Performance Active Networks. In Proc. IEEE Third Conference on Open Architectures and Network Programming, 2000: 37-44

6 Nygren E L, Garland S J, Kaashoek M F. PAN: A High-performance Active Network Node Supporting Multiple Mobile Code Systems. In Proc. IEEE Second Conference on Open Architectures and Network Programming, 1999: 78-89

☆☆

（上接第111页）

简单，但是成功的、有效的。值得注意的是，在我们所研究的两个例子中，单个分类器所采用的识别方法是一样的。对于单个分类器采用不同的识别方法，能否采用上述综合方法进行综合还需要通过实验来论证。

参考文献

1 Park H S, Lee S W. Off-line Recognition of Large-set Handwritten Characters with Multiple Hidden Markov Models. Pattern Recognition, 1996,29(2):231-244

2 Xu L, Krzyak A, Suen C Y. Methods of Combining Multiple Classifiers and Their Applications to Handwriting Recognition. IEEE Transactions on Systems，Man，and Cybernetics, 1992,22(3):418

3 童学锋, 邓 刚, 柴佩琪. 隐马尔可夫模型在脱机手写体汉字识别中的应用. 计算机应用, 2002,22(10):1-3

4 石繁槐, 童学锋. SVM在小字符集脱机手写体汉字识别中的应用研究. 计算机工程, 2002,28(6):154-155

☆☆

（上接第123页）

(3) 结束： 由于只寻找完整的基因，因此只选择以J3'（q_6）结尾的分析，那么最佳的分析的概率就是

$$P^*=\max[\gamma_6(N)]$$

以上是对标准Viterbi 算法的扩展。在具体实现时，首先遍历序列，确定候选的转移位点，然后用上述算法遍历所有的转移。相应的运行时间是O（k^2），k是可能转移的数目。一般而言，k≈n/100，n是碱基的数目。通过对某些状态限制最大序列长度，可进一步减少运行时间，如限制外显子区域长度不超过3000个碱基。

3 小结

该文的算法通过对标准Viterbi 算法的扩展，利用动态编程提高了基因发现的效率。同时，由于HMM模型的模块化和可扩展性，可容易地将上述模型和算法加以扩展，将新的功能单元（状态）加到模型中，以提高基因发现的准确性。对该系统的测试工作正在进一步进行之中。

参考文献

1 Rabiner L. A Tutorial on Hidden Markov Models and Selected Application in Speech Recognition. Proc. of IEEE,1989,77(2): 257

2 Kulp D, Haussler D, Reese M G, et al. A Generialized Hidden Markov Model for the Recognition of Human Genes in DNA. In Proceeding of the Fourth International Conference on Intelligent Systems for Molecular Biology. AAAI Press, Menlo Park, CA,1996

3 Burge C, Karlin S. Prediction of Complete Gene Structure in Human Genomic DNA. J. Mol. Biol.,1997, 268: 78-94

4 Borodorsky M, McIninch J. GeneMark: Parallel Gene Recognition for both DNA strands. Computers Chem.,1993, 17(2): 123-133