对话管理中基于槽特征有限状态自动机的方法研究[1]

黄民烈　　朱小燕

（清华大学计算机系智能技术与系统国家重点实验室 北京 100084）

摘要 对话系统的研究已经成为人机交互技术发展的新热点，而对话管理则是其中最重要的组成部分。本文在当前对话管理的各种实现方法的基础上，提出了一种基于槽特征的自动机设计方法，其中应用了状态压缩和状态集、动作集的子空间划分，并着重以确认过程为例，阐述了确认策略控制函数及其对对话过程的影响。文中还提出了一种树形的意图分层结构，并将这种分层结构应用于主题检测与主题切换，成功解决了多主题对话系统的主题切换问题。最后，实验表明本文提出的设计方案在策略控制、主题检测与主题切换等方面具有较好性能，同时也具有一定扩展性。

关键词 对话系统；人机交互；对话管理；对话策略控制

对话管理模块是对话系统中一个非常重要的组成部分。对话管理的核心内容，就是通过一定的策略控制，指导人机交互顺利进行。通过间接或直接的言语行为，新的对话轮次的发起，对话澄清和纠正，上下文历史记录和语用信息等因素获得相互理解。尤其是在实时语音输入对话系统中，当语音识别错误或者用户提供的信息不完整时，对话管理模块可以对用户进行引导，人机交互得以顺利进行。

在多数较早期的天气、航班、旅馆信息查询对话系统中，如 GALAXY[1]，一般采用填槽法实现对话管理。填槽法把对话过程看作对槽的填充过程，通过不断交互，直至对话目标实现。因为槽相应于数据库中表的条目，所以这种方法也称填表法(Form Filling)，而表的条目也对应语义框架中的格。填槽法所实现的对话过程比较机械，人机交互的自然度较低，但实现复杂度较低，易于开发成熟的商业实用系统。

另一种技术潮流就是有限状态自动机(FSM)方法[2]。这种方法把对话过程看成是自动机的状态转移过程，主要工作是设计自动机的状态和状态转移条件。Kenji Abe 等人把用户和系统看成两个独立的有限状态自动机，把对话过程看成是两个自动机进行信息交换的过程[3]。这种方法虽然思路新颖，但用户模型的不确定性很大，所描述的自动机转移条件过于复杂，状态定义也不甚明晰。和本文的方法相比，上述方法一般针对问题的所有状态进行处

[1]本项目受国家自然科学基金（60272019），（60321002）资助

理，设计每个状态和每条转移弧，没有进行相应状态空间和动作空间的划分，自动机的规模无法控制，而最大的不足在于，缺乏有效的策略控制。在自动机的状态表述方面，Matthias Denecke 提出了用多个信息（如槽被提示的次数，可信度得分，相对频率，语义分析质量，领域一致性等）来引导对话过程[4]。但这些信息如何表述在自动机的状态定义和转移上，原文中没有定义。本文提出的槽特征虽然简单，但较好地利用了可信度得分和对话上下文等信息，而且很好融入状态定义、转移和策略控制中。

近年来，Esther Levin 等人引入了马尔可夫决策过程(Markov decision process, MDP) [5]，将对话过程映射为一个统计模型，并且引入各种算法如增强学习算法用于模型参数的估计。这种方法把对话策略控制看成是在一定代价函数下最优问题的求解过程。但是，这种设计需要上万量级的对话训练样本，样本的获取与标注都非常困难；而且系统设计复杂，学习算法不易收敛，在一般的对话系统应用中，很难实现。本文的研究由于缺乏必要对话资源，只好放弃统计学习方法，但从中借鉴了策略控制函数的基本形式，较好地实现了策略控制。

随着语音技术不断进步和成熟，多主题的对话系统成为研究重点。Bor-Shen Lin 等人把分布式智能 Agent 引入对话管理过程，实现了一个多主题的对话系统[6]。在这种设计中，特定主题的对话模型用一个智能 Agent 表示，主题之间的切换与共享通过 Agent 之间的通讯实现，并有协调器管理各 Agent 之间的协作，在每个 Agent 内部，通过自动机控制状态。这种设计能实现较高的智能性，缺点就是设计复杂度较高，尤其是 Agent 之间的通讯设计，智能体本身也需要较多的对话资源，向相近领域移植需要重新生成 Agent。Xiaojun Wu 等人曾提出了一种主题森林的设计方法[7]，利用与或树将主题组织成树的形式。这种方法在逻辑上比较清楚，但缺乏必要的层次性，而且树的扩展不易进行，对主题的检测机制原文中也没有论及。在这些方法的启发下，本文提出的意图分层树则较好地阐明了层次性、可扩展性和多主题检测机制等关键问题。

总而言之，在前人工作的基础上，本文的改进体现在：其一，在自动机的状态表征方面，提出了用二维槽特征结构表征状态的方法，很好地反映了槽的状态在对话过程中的变化。同时，采用槽的语义分类作为状态表示的组元，而不是具体槽值，有效地避免了状态空间的膨胀，在大规模词表的对话系统应用中意义更加显著。其二，将传统自动机的状态集和动作集进行子空间划分，把自动机的状态转移定义在这些子空间上。这样做可以有效降低自动机的规模，对子空间内的各元素统一控制。在细化的设计中，这些子空间还可以进一步划分，因此也有较好的扩展性。其三，在子空间的内部，究竟如何选取子空间内的元素或者子空间的子集，本文提出了有效的策略控制函数。最后，本文还提出了意图分层结构，并将其应用于策略控制、主题检测与切换中。在本文理论研究的基础上，已开发“家用电器语音控制对话系统”平台－PHAROS 系统。并在后续研究项目的支持下，转换成实用系统投入使用。

本文的后续内容为，第一部分从状态集定义、动作集定义、状态转移及策略控制四个方面介绍了基于槽特征的自动机设计方法。第二部分着重介绍了意图分层结构，并阐述了主题

检测与切换机制。第三部分为实验结果。最后，结束语是工作总结及展望。

1．基于槽特征的自动机设计

在自动机设计方法中，对话过程被看作是自动机的执行过程。对话动作导致了对话状态的变化，同时也使自动机的状态产生变化。状态变化如果达到自动机的终点，则对话结束。这种方法首先设计自动机的状态，然后根据对话流程确定状态之间的转移关系和转移条件。通过不断调整自动机的结构和状态实现对话过程的最优控制，从而使人机交互具有较好的自然度。

为此必须解决如下问题：

（1） 定义状态集 $S = \{s_1, s_2, \cdots, s_n\}$

（2） 定义动作集 $A = \{a_1, a_2, \cdots, a_m\}$

（3） 状态转移控制

（4） 代价函数的表达与求解（策略控制）

本文提出二维结构的槽特征，表征槽的属性和状态。槽特征的定义如下：

$$\begin{bmatrix} slot\_type \\ slot\_state \end{bmatrix}$$

Slot_type 表明槽的类别，即槽值的语义分类，一般人为划分。例如：OBJECT，COMMAND，SHARECMD。值得注意的是，这种特征描述根据槽值的语义类别划分，而不是根据具体槽值。

Slot_state 表明槽值的状态，记录槽值在对话过程中的状态变化。可以取如下值：*UNKNOWN*，*KNOWN*，*VERIFIED*，分别表示槽值在对话过程中的状态为未知状态，已知状态，槽值已被确认的状态。

Slot_state 是状态转移的依据，即转移函数的输入状态，而且它还在确认过程中决定槽值是否需要确认。确认过程中槽值被确认的先后顺序是由 Slot_type 决定的。这些计算过程都与具体槽值无关。而 Slot_type 还在主题检测和切换中有重要应用。因此，这种特征的好处体现在两个方面：其一，采用槽的语义分类而不是具体语义关键词，能够有效地压缩状态空间；其二，特征有效地表达了对话过程中槽的状态变化。

1.1 状态定义

根据二维特征结构，状态的个数可以计算如下：

$$n_{state} = N_{slot\_type} \cdot n_{slot\_state} \tag{1}$$

其中，n_{state} 是状态总数，N_{slot_type} 是语义类别数，n_{slot_state} 是槽值的状态分类数。相比较，如果以具体槽值作为状态的特征描述，状态的个数可以计算如下：

$$\tilde{n}_{state} = \sum_{i=1}^{N_{slot\_type}} n_{slot\_type}(i) \cdot n_{slot\_state} \quad (2)$$

其中 $n_{slot\_type}(i)$ 表示属于第 i 个属性分类的槽值的个数。状态空间压缩率为：

$$\kappa = n_{state} / \tilde{n}_{state} = N_{slot\_type} \Big/ \sum_{i=1}^{N_{slot\_type}} n_{slot\_type}(i) \quad (3)$$

在一般的应用中，属于同一属性分类的槽值通常较多，所以 $\kappa << 1$。由此可见，这种方法的状态空间压缩率是很高的，可以显著降低自动机的规模。

1.2 动作集定义

动作一般指当用户与系统交互时，系统所有可能执行的操作。一般包括与用户的交互（请求输入，直接输出，要求确认等），与外部资源的交互（查询数据库等）及内部处理。

定义动作集合为 $A = A_s \cup A_e \cup A_r \cup A_v \cup A_c$，其中

A_s：表示系统初始化时执行的动作；

A_e：表示系统执行相应语义动作的集合；

A_r：表示向用户询问槽值的集合；

A_v：表示请求用户确认的集合；

A_c：表示对话目标实现时系统执行的动作。

这些子集互不相交，换言之，集合 A_s，A_e，A_r，A_v 和 A_c 是集合 A 的一个完整划分。下面将分别对各个动作子集合进行详细描述。

1． A_s

单元素集合，标记自动机的起始弧，由此系统进入初始状态，表明一次会话过程的开始。例如：

System：欢迎使用 PHAROS 系统。

2． A_e

此集合表示系统执行所有可能的语义动作，为系统内部动作。对用户来说，不直接可见。在 PHAROS 中，此集合的元素是所有家用电器所能执行的动作或者命令，例如“ 电梯开门 ”，“ 打开电视 ” 等。

3． A_r

表示系统向用户询问槽值时的动作集合。其元素组成是所有可能的询问单个槽值的动作。在会话过程中，系统可能执行的这类动作是 A_r 的非空子集，这个子集可能只有一个元素，也可能有多个元素，其值取决于系统决策。

子集只有单元素时，系统针对单个槽进行提问，例如：

[用户已经选择电视机]

System：您想看那个台？

子集有多个元素时，系统提交给用户的问题是对某些槽的组合提问，同时也希望用户在一次对话过程中回应这些槽值，系统决策决定组合的生成。最常见的是针对所有槽提问，例如：

System：您想干什么？

4. A_v

系统请求用户确认时的动作集合。其元素组成是所有可能的确认单个槽值的动作。在会话过程中，系统可能执行的此类动作是 A_v 的非空子集，该子集可能只有一个元素，也可能有多个元素，由系统决策确定。

子集只有单元素时，系统向用户请求单值确认，例如：

System：您是要用电梯吗？

子集有多个元素时，同时针对多个槽值确认，例如：

System：您是想要电梯去二层吗？

5. A_c

系统进入自动机接受状态（Accepting State）对话目标实现后系统执行的动作。为单元素集合，系统执行该动作后，一次会话过程关闭。该动作会在两种情况下发生：其一，系统成功完成用户所要求的查询；其二，会话失败，系统提前关闭会话。例如：

System：谢谢您的使用，再见。

1.3 状态转移

状态集合为 $S = \{s_1, s_2, \cdots, s_n\}$，如前所述，根据状态空间的构成，状态集合可划分为四个子集合：

$$S = \{s_\phi\} \cup S_{un} \cup S_{kn} \cup S_{ve}$$

S_{un}，S_{kn} 和 S_{ve} 分别表示在状态集合中所有二维特征的 slot_state 值为 *UNKNOWN*，*KNOWN*，*VERIFIED* 组成的子集合。s_ϕ 表示 slot_type 和 slot_state 都为空的状态，即初始状态，为单元素集合。必须指出，它与 slot_type 和 slot_state 都为 *UNKNOWN* 的状态是不同的状态。显然，集合 $S_\phi = \{s_\phi\}$，S_{un}，S_{kn} 和 S_{ve} 是集合 S 的一个完整划分，子集合之间互不相交。

由此定义自动机的状态转移函数：

1. $\delta(S_\phi, A_s) = S_{un}$

系统执行 A_s 集合定义的动作后，自动机的状态从初始状态 S_ϕ 转移到 S_{un} 定义的状态。

2．$\delta(S_{un} \cup S_{kn} \cup S_{ve}, A_c) = \{s_0\}$

系统执行 A_c 集合定义的动作后，终止会话过程，自动机回到初始状态。

3．$\delta(S_{kn}, A_v) = S_{un} \cup S_{ve}$

系统执行 A_v 集合定义的确认动作后，自动机从 S_{kn} 状态转移到 S_{un} (确认失败)或者状态 S_{ve} (确认成功)。

4．$\delta(S_{un}, A_r) = S_{un} \cup S_{kn} \cup S_{ve}$

系统执行 A_r 集合定义的请求动作后，自动机状态从 S_{un} 状态转移到 S_{un} 状态(声学解码似然度低)，或者到 S_{kn} 状态(声学解码似然度中等)，或者到 S_{ve} 状态(声学解码似然度高)。

5．$\delta(S_{ve}, A_e) = S_{un}$

系统执行相应的语义动作后(由 A_e 定义)，自动机状态从 S_{ve} 状态转移到 S_{un} 状态。

6．$\delta(S_{kn}, \varepsilon) = S_{kn}$

系统并未执行任何操作(ε 代表空操作)，但用户修改或更正了先前的输入，自动机状态在 S_{kn} 集内跳转。只有 S_{kn} 集合才允许这样的跳转。

如上定义的自动机有两个显著特点：其一，使用了空转移，如6定义；其二，从起始状态作用相同的动作后，到达状态并不唯一，如3和4定义。因此，该自动机是非确定性自动机 $\varepsilon - NFA$（Epsilon Nondeterministic Finite Automata）。

图 1 表示了自动机的大部分转移弧。为了简化，初始状态、起始弧和终止弧没有画出。

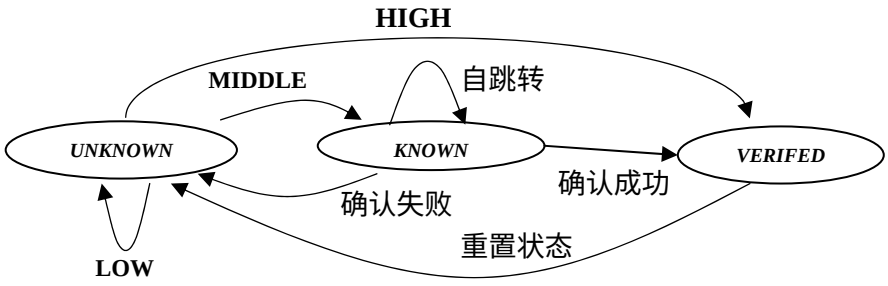

图 1　槽的状态转换图

图 1 中，槽的初始状态为 *UNKNOWN*，如果识别器给出的声学解码似然度（又称可信度）高（HIGH），状态转移至 *VERIFIED*；如果可信度为中（MIDDLE），状态转移到 *KNOWN*，在此状态如果确认失败，状态返回至 *UNKNOWN*，如果确认成功，状态转移至 *VERIFIED*，一次对话成功之后，状态又被重置（如 5 式定义）；如果可信度为 LOW，则状态停留在 *UNKNOWN*。如果处在 *KNOWN*，但是用户对先前的输入进行修改或更正，槽的状态也进行相应更新（自跳转，如 6 式定义），状态仍处在 *KNOWN*。可信度等级由识别器设定相应的高低阈值给出。

自动机状态转移时，需要确定动作 a。选取动作时，采用系统主动的方式 (system

initiative)，即根据当前的状态和对话上下文，系统主动向用户提问为完成对话目标还需完成的槽（将对应格框架的槽填充完全）。系统提问或其它内部行为对应自动机的动作 a，执行相应动作后引起的状态变化对应一个状态转移。系统引导过程也是对话策略控制过程。

1.4 对话策略控制

设对话过程中系统与用户的语句序列为 $M_0, U_0, M_1, U_1, \cdots, M_{i-1}, U_{i-1}$。其中，$M$ 和 U 分别是系统输出和用户输入，i 是对话轮次的标号。当前对话管理的任务可视为求解策略函数：

$$M_i = \pi(M_0, U_0, M_1, U_1, \cdots, M_{i-1}, U_{i-1}; KB) \tag{4}$$

其中 KB 表示知识库。对话管理的策略控制过程可以看作(4)式的求解过程。在本文的研究中，策略控制的任务就是根据当前对话状态，决定系统需要采取的动作类型。

但(4)式无法直接求解。通过引入对话代价函数求解与(4)式等价的解。

一次会话过程（Session）对应状态空间中从初始状态 s_0 到终点状态 s_F 的一条路径，该路径上总的代价称为会话代价（Session Cost），如下式：

$$C = \left\langle \sum_{t=0}^{T_F} c_t(s_t, a_t) \right\rangle \tag{5}$$

T_F 表明对话目标实现时的对话轮次，即对话终点，即 $s_{T_F} = s_F$， c_t 为对话轮次为 t 时的对话代价，s_t 和 a_t 分别表示状态和动作，尖括号表示数学期望。因此，最小化(5)式的过程等价于在相应的代价函数下寻找状态空间中最佳路径的过程。优化该过程就可以实现对话策略控制过程的优化。

最佳策略的求解可以通过(6)式得到：

$$\pi^*(s_t) = \arg\min_a \left[\langle c(s_t, a) \rangle + \sum_s f(s_t \mid s, a) \right] \tag{6}$$

其中 $f(s_t|s,a)$是和转移概率相关的路径代价。求解(6)式的关键在寻找合适的代价函数，但通常很难找到。一般认为，代价函数有如下表达[5]：

$$C = W_i \langle N_i \rangle + W_e \langle N_e \rangle + W_r \langle N_r \rangle + W_o \langle f(N_o) \rangle + W_s \langle F_s \rangle \tag{7}$$

其中,W 系列为权重，表示各成分在整个代价函数中的贡献；N_i 是会话过程中对话轮次的数目；N_e 表示槽值中发生错误的数目; N_r 是从知识库中查询到符合用户查询条件的记录条数，反映了信息获取的代价；N_o 表示系统输出给用户的记录条数，$f(N_o)$ 系统输出记录数目为 N_o 时所花费的代价；F_s 的值为 0 或 1，若有符合查询条件的系统输出，F_s 值为 1，否则为 0。尖括号表示数学期望。

确定(7)式有如下困难：首先各权值难以确定；其次，确定各分项的数学期望需要大量对话样本进行训练，训练算法复杂，不易收敛。

考虑到本文的决策为选取对话动作类型，故提出如下代价函数。为了描述方便，仅以确认过程为例，但对于查询（Request）同样适用。

定义每次只确认一个槽的总代价函数为

$$Q_{total}(n_{slots}, n_{verified}) = \begin{cases} n_{slots} \cdot (1 - p_{one}) + n_{slots} \cdot w_i + 1 \cdot \lambda_{user}, & \text{确认还未开始} \\ n_{slots} \cdot (1 - p_{one}) + n_{slots} \cdot w_i + n_{verified} \cdot \lambda_{user}, & \text{其它情况} \end{cases} \tag{8}$$

其中，n_{slots} 为当前待确认槽的个数，$n_{verified}$ 为按这种方式已确认槽的个数；p_{one} 为这种确认方式下关键词的检出率，一般认为它是一个常数；w_i 为系统完成一次对话轮次所花费代价的权重；λ_{user} 为用户容忍系数，可以大于 1，也可以小于 1，反映用户对这种确认方式的容忍程度。评价函数的第一项表示在单独确认方式下，槽检出的错误率对确认代价的贡献；第二项表示这种确认方式所经历对话轮次的总代价；第三项反映了对话的上下文历史信息。显然，如果需要确认槽个数过多，这种确认方式会影响交互的质量。

(8)式表征的是按单独确认方式完成确认所有槽的总代价，定义平均每次确认的代价为

$$Q_{one} = Q_{total} / n_{slots} = \begin{cases} (1 - p_{one}) + w_i + \dfrac{1}{n_{slots}} \cdot \lambda_{user}, & \text{确认还未开始} \\ (1 - p_{one}) + w_i + \dfrac{n_{verified}}{n_{slots}} \cdot \lambda_{user}, & \text{其它情况} \end{cases} \tag{9}$$

定义一次确认所有槽的代价函数为

$$Q_{all}(n_{slots}) = n_{slots} \cdot (1 - p_{all}) + n_i \cdot w_i \tag{10}$$

其中，p_{all} 表示在一次确认多槽时的关键词检出率，n_i 表示对话轮次的数目，在这种方式下，$n_i = 1$。

决策准则：

$$Decide = \arg\min_{one,all}(Q_{one}, Q_{all}) \tag{11}$$

上式表示，代价函数值较小所对应的确认方式就是系统所决策使用的确认方式。

Q_{one} 随着 n_{slot} 增大而变小（待确认槽数增加时，按 Verify_One 方式平均每次确认的代价降低），Q_{all} 随着 n_{slot} 增大而增大（待确认槽数增加时，按 Verify_All 方式总的确认代价增加），所以 $f(n_{slot}) = Q_{one} - Q_{all}$ 是减函数。其值的正负变化将导致策略的改变。

从基于关键词检出的语音识别引擎的特性来看，句中存在多个关键词时，相互之间的混淆度导致识别率有一定降低，从而导致检出率降低。因此，p_{one}，p_{all} 满足下式：

$$p_{one} \geq p_{all} \tag{12}$$

因此在 λ_{user} 较小，$n_{slot} \leq 3$ 时都有：

$$Q_{one} \leq Q_{all} \tag{13}$$

所以在大多数情况下，应该选取每次确认单个槽值，以获得可靠的系统性能。

上述决策函数较好地反映了对话过程中几个因素：首先是关键词检出率的影响。其次，

由于采用代价的差值进行比较，巧妙避免了 w_i 赋值的影响，而其值很难估计。另外，该决策函数反映了对话上下文的变化。随着对话进行，用户对过于繁琐的提问势必产生逆反心理，从而影响人机交互质量。若继续采用 Request_One 和 Verify_One 的对话方式，将引起对话代价的增大。此外，还综合考虑了用户对 Request_One 和 Verify_One 对话方式容忍程度。如果用户足够宽容与合作（λ_{user} 值很小），系统可以一直采用 Request_One 和 Verify_One 的对话方式，系统表现也更为可靠。相反，如果用户心情急躁，希望尽快完成会话过程，系统可以从开始就采用 Request_All 和 Verify_All 的对话方式，此时，λ_{user} 值相对较大。宏观上分析，这种策略是用户满意度和系统可靠性的折衷考虑。

2．主题检测与切换

所谓主题检测就是在当前对话上下文中，探测对话主题的变更。主题切换就是在当前对话上下文中，从当前的对话主题变更到另一个主题。在多主题对话系统中，新主题的检测和切换是一个显著特点。从应用发展的进程来看，对话系统已经从单领域单主题的应用发展到多领域多主题的应用，开始强调领域之间知识共享和知识相关性研究。在 PHAROS 中，采用基于意图分层树的检测机制，有效地实现了主题切换。这种方法不仅简单、清晰，而且扩展性很好。通过对意图分层树的扩展，可以实现对新主题的支持，而系统结构和控制则不必作任何修改。在本文所实验背景讨论的领域内，定义主题为：一种家用电器的所有操作命令即构成一个对话主题。例如，电视机，电梯都可以成为一个主题。

2.1 意图分层结构

前文提出了两个策略函数，并阐述了多个槽值需要确认或查询时的策略问题。考虑例子：

User：我要用***电梯***，先去***三层***，再回到***二层***。

例中粗斜体为系统处理的关键字。显然，各槽值在对话过程中的重要性并不均等，策略控制必须考虑这种差别。例如，应该尽可能地把带有主题性质的槽（如电梯）先确认，因根据策略控制函数，先确认的槽最可能以 Verify_one 方式进行确认，而这种确认方式更为可靠。为此，本文提出了意图分层的设计方法，系统在查询或者确认之前，依照意图分层结构，对需要确认的槽排序。称之为“ 意图分层 ”，源自用户在输入过程中，其意图在结构上有层次关系，在重要性上也有层次关系，而不是杂乱无章的。下面的对话过程说明了这一特点：

System：欢迎使用 PHAROS 系统

User：我想用电梯。

System:你去哪里？

User:我想去二层。

System：好的，马上去二层。

显然，在此对话过程中，用户第一次输入的“ 电梯 ”表明了一段时间内用户的主要意图，而

后续的对话都是围绕这个主要意图进行的。本文提出意图分层的概念就是为了使得对话过程简单实用、易于控制，同时也符合一般的对话习惯。

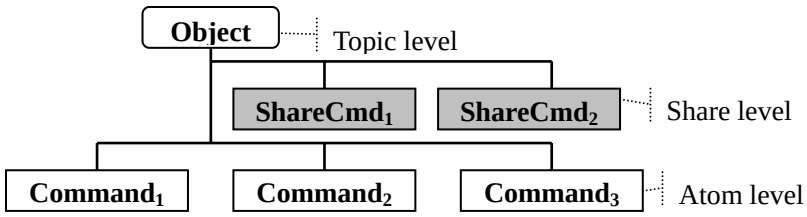

图 2　意图分层树

本文所采用的意图分层结构如图 2 所示。处在最上层的是主题层（Topic Level），这类意图用 OBJECT 类型的关键词标志，如图 3 中的电梯和电视机等。第二层是共享层，可以为各个主题所共享使用的意图层，这类意图用 SHARECMD 类型的关键词标志，例如图 3 中的打开（或开）和关闭（或关）。最下层的是原子层（Atom Level），表明最底层的意图，用 COMMAND 类关键词标志。分层结构自顶向下，一般先经过 Topic Level，Share Level，最后到 Atom Level。但用户也可以不经过 Share Level，从 Topic Level 直接进入 Atom Level，图 2 通过 Topic Level 与 Atom Level 的直接相连表达此含义。多棵树构成一个森林，树与树之间的共享节点即为 SHARECMD 层节点。

图 3 是在 PHAROS 中使用的意图分层树的一个实例。在这两棵树中，“ 电梯 ”和“ 电视机 ”表示所要操纵的两个电器主题，它们共享命令“ 打开/开 ”与“ 关闭/关 ”。此外，“ 电梯 ”的原子命令为“ 上一层 ”,“ 下一层 ”,“ 一层 ”,“ 二层 ”等，“ 电视机 ”的原子命令为“ 中央 1 台 ”,“ 北京 1 台 ”等。值得一提的是，图中每棵树都很容易扩展，可以很方便在主题层、共享层和原子层添加节点。

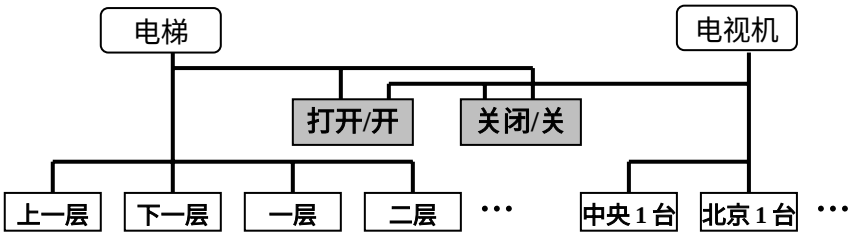

图 3　意图分层树实例

在确认或查询多个槽值时，先根据意图分层树对槽值进行排序。排序的原则是沿树自顶向下，即 topic level 的槽值在最前，share level 次之，atom level 在最后。同一层槽值的先后次序则根据相应关键词的优先级（人工指定）来确定。排好序的槽值经过决策函数的计算，确定系统的动作类型。前述例中的排序结果如下：

Sorted:　　[***电梯***] [***三层***] [***二层***]

2.2 主题检测与切换

根据意图分层结构，很容易实现主题检测与切换。首先定义合法输入：如果某原子项与当前激活的主题在意图分层树中有父子关系，则此输入合法，否则为不合法输入。例如，在

PHAROS 中，如果处在“ 电梯 ”主题下，则合法输入为“ 上一层 ”,“ 下一层 ”,“ 一层 ”,“ 二层 ”等，属于“ 电视机 ”主题的原子项，如“ 中央 1 台 ”和“ 北京 1 台 ”等则为不合法输入。对于非共享层的关键词而言，判断是否合法比较容易。而对于共享层的关键词，由于这些关键词和当前主题在意图分层树上有父子关系，因此没有主题转换的问题，只进行相应的操作。当检测到某个输入关键词与当前主题在意图分层树上无父子关系时，系统提示是否要切换到该关键词所属的主题中去。如果检测到不合法输入，则说明用户可能想更换主题，此时系统应提示用户是否确定要改变主题，或者系统自动切换主题。因此，PHAROS 把非法输入当作切换主题的触发点。

语义框架中各个格实例化时与意图分层树中各种节点的对应关系如图 4。语义框架中有三个关键的格：*AGT*，它被 Object 类型的关键词实例化，表明对话主题；*OBJ_AUX*，辅助动作格，由 SHARECMD 类型的关键词实例化，在 PHAROS 中，如“ 打开/开 ”和“ 关闭/关 ”；*OBJ*，标明原子项，指具体的动作，由 COMMAND 类型的关键词实例化，在图 4 中，槽 *OBJ* 可以被从 $Command_{11}$ 到 $Command_{1n}$ 的各个节点实例化。

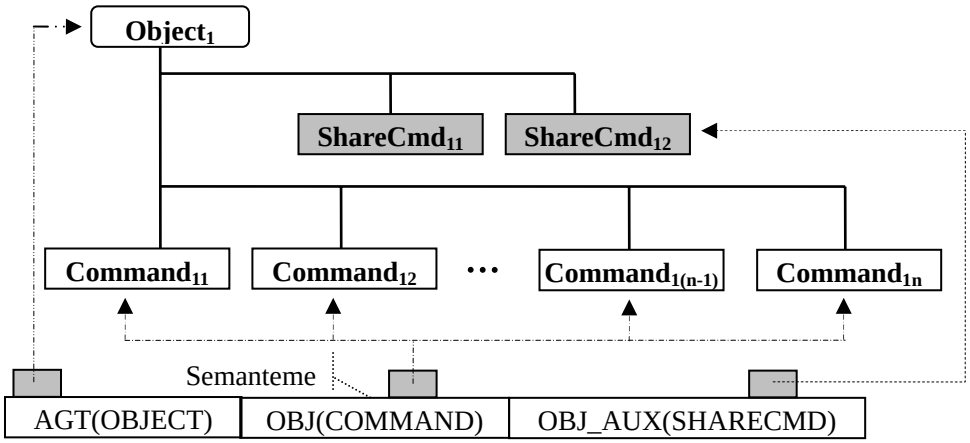

图 4　语义框架的格与意图分层树节点对应关系图

如果输入不合法，在某个语义框架中的格实例化后，还需找到与这个输入相应的主题，即待变更的主题。对于某个槽值(假设为 *slot-value*)，搜索算法如下：

1．得到当前主题 *curTopic* 及相应的意图分层树 *curTree*；

2．判断是否合法输入，如果是，则返回；

3．Case　slot_type == OBJECT:

　　If *Slot-value* != *curTopic*

　　Prompt user to switch topic;

　　else

　　other processing;

4．Case　slot_type == SHARECMD:

　　If *slot-value* $\notin$ *curTree*

　　FindMostRelevantTopic(*curTree*, SHARECMD, *slot-value*);

Prompt user to switch topic;

Else

Other processing;

5． Case slot_type == COMMAND:

If *slot-value* ∉ *curTree*

FindTopic(*curTree*, COMMAND, *slot-value*);

Prompt user to switch topic;

Else

Other processing;

算法中，对于 COMMAND 类型的槽值，在除 *curTree* 之外的其它意图分层树的 atom level 层节点中搜索，直到找到一个与槽值相同的节点，由函数 *FindTopic* 实现。因为每棵树的 atom level 节点各不相同，所以对于 COMMAND 类型的槽值，只可能找到一棵树与之对应。但对于 SHARECMD 类型槽值则有所不同。因为 share level 的节点可能被若干棵树同时拥有，因此返回结果可能是多个主题，这时不能简单随机选取。为了排除这种二义性，算法用堆栈保存了对话过程中所有未完成的对话主题。堆栈保证了最近的未完成主题最先出栈，符合一般对话习惯。若搜索返回的候选结果在堆栈中存在，则从栈中弹出最靠近栈顶的主题作为结果。如果任何候选都不在栈中，则根据其他上下文信息（如最近 5 个关键词的信息，所有已完成的主题信息等）选择一个最相关主题。这些功能由 *FindMostRelevantTopic* 函数实现。搜索只发生在所有意图分层树的某一层上，因此即使树的数目很大，搜索效率也很高。

3．实验分析

实验分三个部分进行：关键词检出实验，策略函数实验，主题检测与切换实验。

3.1 关键词检出实验

由公式(9)、(10)和(11)可知，语音识别引擎的性能直接影响策略控制过程。因此，首先进行识别引擎的性能测试。本实验中，采用本课题组研制的用于关键词检出的连续语音识别引擎。测试语料为两类，一类为只含有一个关键词的句子共 100 句。另一类为含有两个或两个以上关键词的句子，也是 100 句。检出率又分为第一关键词检出率和第二关键词检出率。计算公式分别为：

$$accuracy1 = \frac{\text{正确检出第一个关键词 的个数}}{\text{句子包含的第一关键词 总数}} \tag{14}$$

$$accuracy2 = \frac{\text{正确检出第二个关键词个数且第一个关键词检出正确}}{\text{句子包含的第二关键词总数}} \tag{15}$$

表 1 和表 2 给出了实验结果。

表 1 关键词检出率实验一

单个关键词（数据组号）	句中关键词总数	检出数目	关键词检出率

accuracy1	100	95	95%

表 2 关键词检出率实验二

多个 关键词	句中 第一 关键词 总数	第一 关键词检 出 数目	句中 第二 关键词 总数	第二 关键词检 出 数目	第一 检出率	第二 检出率
accuracy2	266	245	266	230	92.1%	86.5%

表 2 中第一检出率比表 1 中检出率低，是因为句子中含有其它关键词，使得整个句子的混淆度增加所造成的。

表 3 是实际系统 PHAROS 中所使用的关键词表的基本情况。

表 3 关键词表

关键词语义分类 (Slot_type)	关键词实例	关键词 数目	总数
Object	电梯、电视机(电视)[1]、电脑(计算机)[1]、轮椅，智能声控设备等	20	约 150
Share-command	开,关，开门，关门，打开，关闭，播放，暂停，继续，开始，停止[2]	25	
command	一层，二层，三层；换台，中央 1 台，中央 2 台，…；重启，注销，上网，收 email；前进，后退等	约 105	

注 1：括号中表示同义词，但在关键词数目中不会重复计数；

注 2：此类指两两主题的共享，也有个别主题与任何其它主题都没有共享；

注 3：表 1 和表 2 实验中关键词只使用了表 3 中的约 100 个关键词。

3.2 策略函数实验

在本实验中，根据检出率实验，取 p_{one} ＝0.95， p_{all} ＝0.865。假设初始待确认的槽数为 5。图 5 给出了 λ_{user} 分别取 0.2、0.9、1.2 和 2.4 时，函数值 $f(n_{slot})=Q_{one}-Q_{all}$ 随当前待确认槽个数的变化情况。

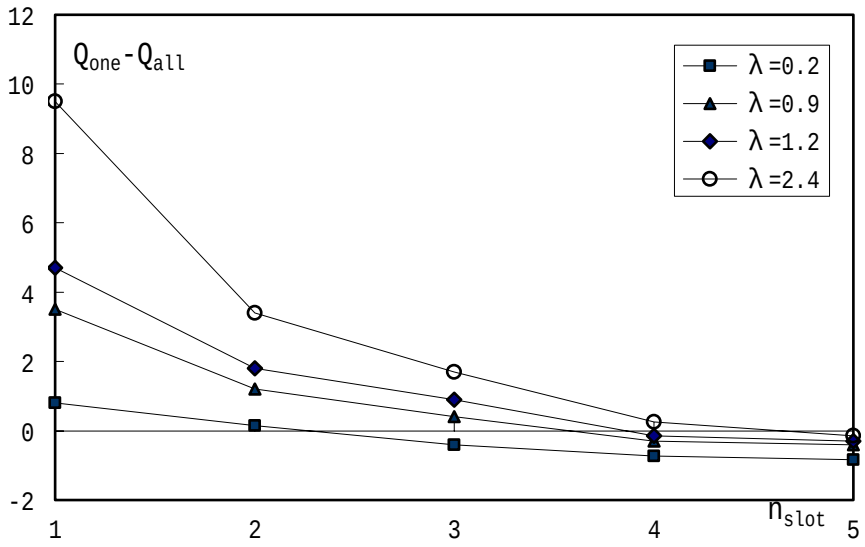

图 5　$f(n_{slot})=Q_{one}-Q_{all}$值在确认过程中的变化曲线

函数值在 $n_{slot}=5$ 都小于零（确认还未开 始），表明 λ_{user} 足够小时在开始确认阶段倾向选取 Verify_One 方式；$\lambda_{user}=2.4$，$n_{slot}=4$ 时开始大于零（按 Verify_One 方式确认的代价比按 Verify_All 方式确认的代价大），根据决策准则(11 式)，表明按 Verify_one 方式确认完一个槽值后就开始选取 Verify_All 方式；而 $\lambda_{user}=0.9$，1.2 时，函数值 f 在 $n_{slot}=5$ 和 $n_{slot}=4$ 时都小于零，在 $n_{slot}=3$ 才开始大于零，表明在按 Verify_one 方式确认完两个槽值后，才开始选取 Verify_All 方式；$\lambda_{user}=0.2$ 时，函数值 f 在 $n_{slot}=2$ 才开始大于零，表明在按 Verify_one 方式确认完确认完三个槽之后才开始采用 Verify_All 方式。

由此可见，λ_{user} 表明用户选择两种方式的倾向程度：λ_{user} 越大，用户越倾向选择 Verify_All 方式；相反，λ_{user} 越小，用户越倾向选择 Verify_one 方式。下面两个对话测试实例分别是 $\lambda_{user}=0.9$ 和 $\lambda_{user}=2.4$ 时系统所引导的对话过程：

实例 1：　System：欢迎使用 PHAROS 系统。

User：我想要电梯先到三层，然后回到二层，喔，再下一层吧。

System：您是要用电梯吗？

User：是的。

System：是先到三层吗？

User：是的。

System：然后回到二层，最后到一层？

User：是的。

实例 2：　System：欢迎使用 PHAROS 系统。

User：我想要电梯先到三层，然后回到二层，喔，再下一层吧。

System：您是要用电梯吗？

User：是的。

System：是先上到三层，然后回到二层，最后到一层吗？

User：是的。

从实例 1 和实例 2 中的对话过程可以看出，该参数直接导致了对话过程中策略变化。实例 1 在以 Verify_one 方式确认完两个槽值后（电梯，开门）开始采用 Verify_all 方式进行余下的确认，而实例 2 则在以 Verify_one 方式确认完一个槽值后（电梯）就开始采用 Verify_all 方式进行余下的确认。因此可以根据用户需求设定更加符合用户习惯的用户模型。

上述过程基于确认过程，但对询问同样适用。因论文篇幅原因，在此省略。

3.3 主题检测与切换实验

实验分两种情况来进行：一种非法输入为 Topic Level 的关键词，另一种非法输入为 Atom Level 的关键词。分别见实例 3 和实例 4。

实例 3： System:欢迎使用 PHAROS 系统

User：我想用电梯。

System：你去哪里？

User：还是回去看电视吧。

System：对不起，你是想看电视吗？

User：是的。

System：想看哪个台？

实例 4： System:欢迎使用 PHAROS 系统。

User：我想用电梯。

System：你去哪里？

User：我想看中央 1 台。

System：是要看电视吗？

User：是的。

System：请欣赏中央 1 台。

在实例 3 中，系统已经在“ 电梯 ”状态，但在第三轮检测到 Topic Level（电视）的非法输入，于是系统提示用户是否切换主题。如果用户输入肯定答复，则切换到电视机状态。实例 4 中，在第三轮检测到 Atom Level 的非法输入（中央 1 台），提示用户是否切换主题；若是，不仅切换主题，系统还执行了先前 Atom Level 的输入（中央 1 台）。

4．结束语

本文提出了一种基于槽特征的改进型自动机方法，并实现了策略控制函数，较好地解决了对话管理中的策略控制。策略控制函数不仅较好地反映了识别率（或者槽的检出率）的影

响，也反映了对话过程中对话轮次对交互质量的影响，还反映了对话上下文的变化及用户对不同对话方式的容忍程度。此外还提出了意图分层结构，并将其用于槽值排序及多主题检测与切换。实验表明将意图分层结构用于主题检测与切换，不仅简单而且非常有效。

本文所讨论的实验背景虽然只涉及相对简单的应用领域，但本文所提出的方法具有一定推广性和普遍性。例如在一个查询航班、天气情况和旅馆住宿情况的多领域对话系统中，本文基于槽特征的方法对状态压缩和控制自动机的规模同样适用，而策略函数无疑也是可行的。意图分层树也适用于主题检测和切换，只是需要修改树的结构，树的节点数目也可能更多。为了实现特定需求，分层树可设计成多于三层，层与层之间的优先顺序可任意安排（不一定是 top-down），树与树之间可以有共享节点也可无共享节点。这种机制简单有效，容易实现，具有一定扩展性。

但仍存在问题没有解决。例如，参数是事先固定的。显然，不同用户的参数应该有所区别。虽然可根据用户的需求事先设定，但仍存在先入为主的问题。因此我们计划在后续研究中通过用户模型的生成与更新，自动获取用户参数。同时，p_{one} 和 p_{all} 参数也依赖于语音识别引擎的性能，与用户对语音识别引擎的适应程度有关，我们计划进一步研究如何根据用户的特性对这两个参数进行自适应修正。

参考文献

[1] Goddeau D., Brill E., Glass J., et al. GALAXY: A human language interface to online travel information. In: proceedings of the International Conference on Spoken Language Processing (ICSLP’94), Yokohama, Japan, 1994, pages 707-710

[2] F. Huang, J. Yang, A. Waibel. Dialogue management for multimodal user registration. In: proceedings of the International Conference on Spoken Language Processing (ICSLP’2000), Beijing, China, October, 2000, vol.3, pages 37-40.

[3] Kenji Abe, Kazushige Kurokawa, Kazunari Taketa, Sumio Ohno, and Hiroya Fujisake. A new method for dialogue management in an intelligent system for information retrieval. In: proceedings of the 6th International Conference on Spoken Language Processing (ICSLP’2000), Beijing China, 2000, pages 118-121

[4] Matthias Denecke. Informational characterization of dialogue states. In: proceedings of the International Conference on Spoken Language Processing (ICSLP’2000), Beijing, China, 2000, vol.2, pages 114-117.

[5] Esther Levin, Roberto Pieraccini, Wieland Echert. A stochastic model of human-machine interaction for learning dialog strategies. IEEE Transactions on speech and audio processing, January 2000, vol.8, No.1: pages 11-23

[6] Bor-Shen Lin, Hsin-min wang, and lin-shan Lee. A distributed agent architecture for intelligent multi-domain spoken dialogue systems. IEICE Trans.Inf&Syst., September 2001, Vol.E84-d, No.9:

pages 1217-1230
[7] Xiaojun Wu, Fang Zheng and Mingxing Xu. Topic forest: a plan-based dialogue management structure. In: proceedings of IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP2001), Salt Lake City, USA, May 2001, pages 617-620

A Finite State Automata Approach Based on Slot-feature for Dialogue Management in Spoken Dialogue System

HUANG Min-Lie ZHU Xiao-Yan
State Key Laboratory of Intelligent Technology and Systems,
Department of Computer Science and Technology, Tsinghua University, Beijing, China, 100084

Abstract Spoken dialogue systems have become a new hotspot of human-machine interaction technology, in which dialogue management is most important. In this paper, we propose a new method for dialogue management. The scheme uses finite state automata based on slot-features to establish the whole structure. In the automata, a two-dimension state feature is proposed and state reduction is applied. State set and action set are divided into subsets; hence it becomes easier to control. The most distinctive aspect is that two strategy control functions are used to implement the strategy control in the verification as well as in the request. Furthermore, an intention-layered structure is proposed and applied to the topic detection and switch, which functions for multi-topics applications. The structure is portable and feasible to be extended. At last, experiments show that our method has a good performance.

Keywords spoken dialogue system; human-machine interaction; dialogue management; dialogue strategy control

作者简介

黄民烈，男，1977 年 12 月生，博士研究生，研究方向为语义理解，人机交互模型，信息抽取和 web 信息检索

HUANG Min-Lie, born in December 1977, Ph. D. candidate. His research interests include language understanding, human-machine interaction model, information extraction and web information retrieval.

朱小燕，女，1982 年获北京科技大学学士学位，1987 年获日本神户大学硕士学位，1990 年获日本名古屋工业大学博士学位。1993 年到清华大学任教，现为清华计算机系教授，博士生导师。主要研究领域为智能信息处理，其中包括：模式识别；神经元网络；机器学习；自然语言处理；信息提取等。

ZHU Xiaoyan, born in 1957, professor, Ph. D. supervisor，Deputy Head of Department of Computer Science and Technology, Tsinghua University. She got bachelor degree at Beijing Science and Technology University in 1982, master degree at Kobe University in 1987. and Ph. D. degree at Nagoya Institute of Technology, Japan in 1990. She is teaching at Tsinghua University since 1993. Her research interests include pattern recognition, neural network, machine learning, natural language processing and information retrieval.

联系人电话：黄民烈 13520566808 huangml00@mails.tsinghua.edu.cn or huangminlie@tsinghua.org.cn

Background introduction

The work described in the paper is a sub-task of the research project which was supported by Chinese Natural Science Foundation with the contract No.60272019. This research project is contributed to model an understanding-based human-machine interaction, which consists of multi-mode HCI (including speech, keyboard, and scanned image), feasible user model establishment, and natural, intelligent spoken dialogue system. Besides, the project also aims to implement an intelligent human-machine interaction system with feasible interacting interface, self-learning and self-adaptive capability. We have done much research on robust speech recognition and verification, language understanding, semantic analysis, user model and spoken dialogue system. Several papers have been published and a spoken dialogue system to control

electronic devices such as elevator, TV set, is implemented for real application. This paper focuses on dialogue management of the spoken dialogue system to provide user a natural, intelligent interaction with computer, through the help of feasible dialogue strategy control and multi-topics detection and switch.