ISSN1000-9825 Journal of Software 软件学报 2000,11(2):277~284

文本规划中焦点移动控制算法的研究*

姚天昉 汤学彦

(上海交通大学计算机科学与工程系 上海 200030)

E-mail: yao-tf@cs.sjtu.edu.cn

摘要 该文完善了全局焦点和局部焦点的移动规则,使规则适合更广泛的应用领域.所提出的新算法以 Mckeown 的焦点算法为基础,将原来的栈式控制结构改进成树型控制结构,可以同时控制全局焦点和局部焦点的移动.新算法改进了 Schema 的扩充转移网络表示,为弧增加了重复特性和优先级的信息,还加入了对 Schema 进行递归的控制和回溯机制,并由知识库决定 Schema 的填充和递归.此外,又增加了对 Schema 可交换符号的处理.在此基础上,进一步介绍了实验系统的设计,并讨论了算法的效果.

关键词 文本规划,全局焦点,局部焦点,模式,谓词.

中图法分类号 TP18

文本生成不仅需要保证句子语义、句法结构和词法的正确性,而且必须围绕文本的主题来组织句子,使它们恰当地衔接,以保证文本的连贯性.一般说来,文本生成可分为 3 个阶段,即文本规划(text planning)、句子规划(sentence planning)和表层生成(surface generation)[1].焦点控制机制是文本规划中的一种重要的机制,它在组织文本内容时跟踪两个相邻句子或者段落之间焦点的移动,以确保所生成的文本是连贯的.

Schema 方法是采用语言学中的修辞性谓词(rhetorical predicate,简称 predicate)来表达文本结构的一种方法.文本是由命题(proposition)组成的.Proposition 是一个句子或者一个从句.Predicate 将文本中的 Proposition 分类,每个 Proposition 都被归纳为特定的 Predicate.对同一类型文本,存在着一些标准的 Predicate 的组合模式来表示文本的结构,这种标准模式称为 Schema.用 Schema 表示的文本结构是一种层次化的结构,即 Schema 可以递归嵌套[2].在文本生成阶段,先选择合适的 Schema,然后用知识库中的内容填充匹配 Schema 中的 Predicate,为 Predicate 的参数赋值.每个 Predicate 都包含若干参数,参数的个数和类型是由其语义决定的[3,4].Proposition 是将 Predicate 中的参数赋值后所得到的实例.Schema 的填充过程是在遍历其 Predicate 的同时,利用 Predicate 的语义选择适当的 Proposition 进行匹配.当出现多种匹配方案时,由焦点控制机制决定选择哪个 Proposition.最后输出遍历 Schema 所得的 Proposition 序列,供后续阶段使用.

文本是围绕某些概念或者对象的描述展开的,它们称为焦点.焦点控制机制限制所生成文本的内容,使整个文本尽可能地连贯.文本中存在着两种类型的焦点:全局焦点(global focus,简记为 GF)和局部焦点(immediate focus,简记为 IF).GF 描述的是文本段落或者文本的表述中心,与 Schema 一一对应.为了使文本的结构更清晰,我们假设所涉及文本的 GF 只有一个.GF 只有在 Schema 发生递归时才可能发生变化.文本生成之前,必须提供最外层 Schema 的 GF.如果 Schema 出现递归,在内层 Schema 填充结束后,恢复外层 Schema 及其 GF,继续填充.GF 的控制与 Schema 递归密切相关.Schema 递归实现的关键在于如何决定被嵌套的新的 Schema 的 GF.IF 是指一个句子所表述的中心概念或对象,是 Proposition 中一个被赋值的参数.Proposition 的赋值参数可以分为两类,一类是可能成为焦点的参数,另一类是不可能成为焦点的参数.为了控制 IF 的移动,在文本生成过程中,

* 本文研究得到国家自然科学基金(No.69673008)、上海市科委科技发展基金(No.96297002)和德国大众基金(VW)资助.作者姚天昉,1957 年生,副教授,主要研究领域为机器翻译,自然语言生成.汤学彦,1977 年生,硕士生,主要研究领域为自然语言生成,计算机网络.

本文通讯联系人:姚天昉,上海 200030,上海交通大学计算机科学与工程系

本文 1998-11-23 收到原稿,1999-03-01 收到修改稿

需要跟踪和维护下列3个方面的信息：

(1) 当前句子的IF(称为当前焦点,简记为CF)；

(2) 当前句子中可能成为下一个句子的IF的成份(称为potential focus list,简记为PFL)；

(3) 前文中部分句子的IF所构成的焦点栈(focus stack)[2].

本文主要研究基于Schema技术的文本规划中的焦点控制机制.

1　焦点移动规则

研究IF的重点在于如何在相邻两个句子之间变换或者保持焦点.Sidner提出4条焦点移动规则[2]:新的CF可以是:① 前句PFL中的元素;② 前句的CF;③ 从焦点栈中弹出的元素;④ 与前句CF隐含相关的概念.Mckeown将上述规则中的前3条应用于焦点算法中[2].如果只使用前3条规则,那么CF转移至一个从未成为过焦点的新概念时只能通过规则①,而所有PFL中的元素都被前一个句子提及.所以,当一个概念没有被提及时,它就不可能成为焦点,因此其通用性较差.例如:

上海中心气象台7月24日21点钟发布上海市天气预报(Focus=天气预报):今天夜里和明天上海市和长江口区天气预报(Focus=天气):阴有时有阵雨(Focus=天气),局部地区雨量中等(Focus=雨),明天转阴到多云(Focus=天气),偏南风(Focus=风),明天转东南风(Focus=风),风力都是4到5级(Focus=风),明天最高温度33度(Focus=温度),明天最低温度26度(Focus=温度).

在上述天气预报文本中,CF由天气转入风、温度时,必须应用规则④.但规则④中CF的转移范围太广,必须加以一定的限制.在分析文本的基础上,我们将规则④修改为"涉及一个与当前Schema的GF隐含相关的概念(implicitly related focus,简记为IRF)".新的规则④只允许CF转移至IRF,以避免使文本结构过于松散.在上例中,风和温度都是与天气预报密切相关的概念.记IRF(Object)为与对象Object隐含相关的对象的集合.下面来定义新的IF移动规则:

规则1. 焦点转移为前句的潜在焦点,即CF(new sentence)∈PFL(last sentence).

规则2. 焦点保持前句的当前焦点,即CF(new sentence)=CF(last sentence).

规则3. 焦点回到先前讨论的某一焦点,即CF(new sentence)∈Focus-Stack.

规则4. 焦点转移到与全局焦点隐含相关的新焦点,即CF(new sentence)∈IRF(GF).

这里需要说明的是,上述4条规则描述的是一个Schema(对应于一个GF)内部IF的变化,不允许跨Schema使用.为了利用这些规则,必须为它们赋以优先级.根据文本连贯性的特点,并结合对文本的分析,我们提出下列优先级关系:

Priority(规则1)>Priority(规则2)>Priority(规则3)>Priority(规则4)

文本生成时若存在多个Proposition可供选择,则需要利用焦点移动规则进行控制.首先删除不符合任何一条上述规则的Proposition,然后按优先级从高到低依次运用这些规则,选择符合最高优先级规则的Proposition.

2　Schema的实现方法

Schema是通过扩充转移网络(augmented transition network,简称ATN)的方法实现的.ATN原来在自然语言理解中用来分析句子[5].代表Schema的ATN是一张有向图,它的结点代表起始状态、填充过程中的中间状态或终止状态.Schema的递归嵌套对应于ATN的递归嵌套.ATN弧共有两种类型.一种是Schema/Predicate,代表Schema中的一个符号,包括它的Schema和Predicate实现,如果该符号不能扩展成另一个新的Schema,则Schema实现为null.另一种是jump,代表状态之间的无条件转移,即转移时无需匹配任何Schema或Predicate.在每条弧上设置一个代表重复次数的特性.对Schema/Predicate型弧,此信息表示Schema和Predicate实现可以重复的次数,"1"表示1次,"+"表示任意多次(即1～N次).对jump型弧,重复次数一般为1次.在Schema被填充的过程中,如果匹配失败就需要回溯,退回到先前的状态,选择其他的Predicate进行匹配.所以,为每条弧还增加了一个优先数,优先数越小,优先级越高.在转移到某一状态后,先选择以该状态为弧尾的弧

中优先级最高的(不一定是一条)进行匹配.若因填充不成功而回溯到该状态,再选择下一优先级的弧进行匹配,依此类推.弧可以用如下四元组表示:(Predicate 实现,Schema 实现,重复特性,优先数).对于 jump 型的弧:Predicate 实现=jump,Schema 实现=null,重复特性="1".表 1 为 Constituency Schema,其对应的 ATN 如图 1 所示.

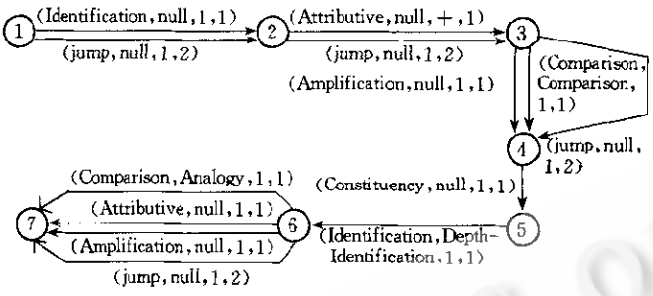


Fig. 1 ATN of constituency schema
图1 Constituency schema所对应的ATN

Table 1 Constituency schema
表1 Constituency 模式

Constituency Schema
{Identification}
Attributive*
{Comparison/Amplification}
Constituence
{Depth-identification}+
{Attributive/Analogy/
Amplification}

这里的 ATN 与 Mckeown 表示 Schema 的 ATN 有所不同.主要区别是:

(1) 在 Schema 的 ATN 表示中不允许出现了图(即 Mckeown 的 ATN 中的 subr).子图的作用是提取 Schema 表达式中可共享的子表达式,用另一个 ATN 表示,以简化 Schema 的表示.在填充 Schema 过程中,若转移到子图进行匹配时,CF 可以转移到外层 ATN 中已匹配所得的 Proposition 的 CF(规则 3).若转移到被嵌套的 Schema 所对应的 ATN,则不允许出现这种情况,因为 IF 移动规则不允许跨 Schema 使用.所以两者的焦点控制是不同的.在这里,Schema 与 ATN 一一对应.控制在 ATN 之间的转移只可能是由于 Schema 的递归造成的,原因是便于控制.

(2) 为每条弧增加了重复特性的信息.在 Mckeown 的 ATN 中,通过环来表示 Schema 或 Predicate 实现重复任意次数.遇到此类情况,需要多次匹配,每次只从匹配结果中选取一个 Proposition.在改进后的 ATN 中,Schema 或 Predicate 重复任意次数既可以用上述方法实现,也可以用一条弧(重复特性为"+")实现.如果用后一种方法实现,只需一次匹配便可输出多个 Proposition,便于对它们按照时间、空间等顺序排序.对这两种方法,IF 的移动情况是不同的.如果用环实现,那么同一 Predicate 所对应的多个 Proposition 之间的 IF 允许通过规则 1 或规则 3 转移.如果用"+"型弧实现,那么这些 Proposition 之间的 IF 相对于前面的句子来说是一种并列关系.应该根据具体情况选择合适的方法.例如,介绍事物属性的 Attributive 谓词就适宜用后一种方法实现.因为 Attributive+着重对事物的属性加以介绍,中间 IF 不应该转移.否则可能出现下列情况,使文本结构变得松散而不连贯(文本中的黑体字表示连贯部分).

弹道导弹型核潜艇是一种战略武器(Focus=弹道导弹型核潜艇).弹道导弹是装有弹头和动力装置并能制导的高速飞行武器(Focus=弹道导弹).弹道导弹的射程较远(Focus=弹道导弹).弹道导弹的飞行轨道是椭圆曲线(Focus=弹道导弹).**弹道导弹型核潜艇在首部设有 4~6 个鱼雷发射管**(Focus=弹道导弹型核潜艇).**弹道导弹型核潜艇一般携带 12~16 枚核弹头的弹道导弹**(Focus=弹道导弹型核潜艇).**弹道导弹型核潜艇长期在水下活动,出没无常,难以发现**(Focus=弹道导弹型核潜艇).

3 焦点控制算法

为了加入对 IF 移动规则 4、Schema 递归以及填充过程中回溯的处理,我们改进了 Mckeown 的焦点算法中的栈式控制结构,采用树型控制结构(称之为焦点控制树).这样做的目的主要有两个:① 原来的栈式结构不适合处理 Schema 递归,而树是一个具有递归定义的数据结构.如果将一个 Schema 对应文本按焦点的转移规律组成一棵树,那么 Schema 的递归可以看成是将树中的结点扩展成一棵新的子树,这样可以方便地实现递归;② 采用树型控制结构可以保留迄今为止产生的所有文本及其焦点信息,便于进行回溯.在栈式控制结构中如果执行

了出栈操作,则很难在回溯的时候恢复以前的状态.在树型控制结构中填充Schema的过程就是焦点控制树不断生长的过程,在回溯的时候只需回到先前生成的某一结点,删去其子树即可.算法中的回溯分为两个阶段.第1阶段是在规则1和规则2不能适用时,通过回溯寻找焦点栈中符合规则3的元素.如果找到,则匹配相应的Proposition,否则回溯到当前填充Schema对应的子树的根结点,利用规则4作筛选.如果还不能找到合适的Proposition,则恢复第1阶段回溯以前的状态.在第2阶段回溯时,逐步删除已经构建好的焦点控制树,选择ATN图中的其他候选Predicate进行匹配.第1阶段的回溯不允许跨Schema进行.第2阶段的回溯允许跨Schema进行,可以将原先采用Schema实现填充但没有成功的符号改用Predicate实现匹配,继续尝试.

选用二叉树作为控制结构.每个结点的左子树表示与该结点具有相同CF的结点,右子树表示由具有该结点的CF引伸出的新焦点的结点或者是被嵌套的Schema所对应子树的根结点.二叉树中的每个结点可能对应文本中的一个Proposition,也可能是为了构造焦点控制树的需要而虚设的结点,不对应任何实际的Proposition.每个结点都表示填充Schema过程中的中间状态,包含如下信息:① 当前正在填充的Schema,包括其对应的ATN、子树的根结点和GF;② Schema对应的ATN中到达的状态;③ 匹配所得的Proposition,包括Proposition的CF,PFL等;④ 回溯时的前驱结点,如果前驱结点是父结点,则以-1表示;⑤ 若是Schema所对应子树的根结点,还需记录当前进行的匹配是按照Schema实现扩展填充还是Predicate实现进行匹配.回溯时,对Schema扩展填充将其改用Predicate实现继续尝试,对Predicate匹配,则进一步向前回溯.所以,对于ATN中的每个状态,实际匹配时进行选择的优先权从高到低依次为:最高优先权弧的Schema实现,最高优先权弧的Predicate实现,次高优先权弧的Schema实现,次高优先权弧的Predicate实现,...,依此类推.

对Schema中匹配的第1个Proposition,焦点栈为空,故没有前句的PFL可供选择,所以规则1和规则3都不可能使用.此时,CF可以是该Schema的GF(规则2的推广)或者其IRF中的元素(规则4).在算法中,在每个Schema对应了树的根结点中置CF—GF,其他Proposition信息为null.这样,在匹配Schema的第1个Proposition时就只能使用规则2和规则4了.下面给出按照4条IF移动规则选出Proposition后对焦点控制树的处理.以C表示控制树中的当前结点,N,N_1,N_2,...表示新增加的结点.水平链表示二叉树的右子树,垂直链表示二叉树的左子树.Schema,Status,Proposition,Prec和Match域分别表示结点的上述5类信息.

算法1. 选出符合IF移动规则1的Proposition后对焦点控制树的处理.

① 按照表2扩展焦点控制树;② 如果被匹配的弧有Schema实现,置$C=N_1$;否则置$C=N$.

算法2. 选出符合IF移动规则2的Proposition后对焦点控制树的处理.

① 基本与算法1的步骤1相同,不同之处是,当被匹配的ATN弧没有Schema实现且重复特性为"1"时,新增加的结点N是当前结点C的左子树而不是右子树;② 如果被匹配的弧有Schema实现,置$C=N_1$;否则置$C=N$.

算法3. 选出符合IF移动规则3的Proposition后对焦点控制树的处理.

在第1阶段回溯的时候应用规则3.设第1阶段回溯前的当前结点为C,在第1阶段回溯时,从当前结点开始沿着二叉树向祖先方向寻找能应用规则3的结点.设找到的结点为F.

① 按表3扩展焦点控制树,目的是建立第2阶段回溯的Prec(前驱结点)链,并置$C=P$;② 与算法2的步骤1相同,但所有新增结点的Prec域都置为C;③ 如果被匹配的弧有Schema实现,置$C=N_1$;否则置$C=N$.

算法4. 选出符合IF移动规则4的Proposition后对焦点控制树的处理.

设第1阶段回溯前的当前结点为C.在第1阶段回溯结束尚未找出合适的Proposition的时候应用规则4,此时已回溯到当前填充Schema对应子树的根结点.

① 从该根结点开始沿左子树链寻找最深层的结点,设为F;② 以下同算法3.

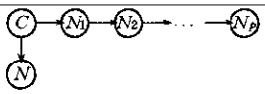
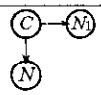
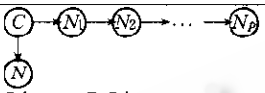
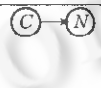
下面给出填充Schema时完整的焦点移动控制算法.

算法5.

① 初始化:设置根结点Root,其中Root.Schema=待填充匹配Schema,Root.Status=待填充匹配的Schema所对应的ATN,Root.Proposition=null,Root.Prec=-1,Root.Match=PredicateMatch,C=Root;

Table 2 Processing for controlling tree of focus in algorithm 1

表2 算法1对焦点控制树的处理

	被匹配的ATN弧重复特性为"+"	被匹配的ATN弧重复特性为"1"
被匹配的ATN弧有Schema实现	 其中N. Schema=C. Schema; N. Status=当前ATN匹配Proposition到达的新状态; N. Proposition=null, N. Prec=−1, N. Match=PredicateMatch; N_i. Schema=该弧的Schema实现; N_i. Status=该弧Schema实现的ATN起始状态; N_i. Proposition=选中的各Proposition的信息; N_i. Prec=C, N_i. Match=SchemaMatch. $i=1,2,\ldots,p$	 其中N. Schema=C. Schema; N. Status=当前ATN匹配Proposition到达的新状态; N. Proposition=null, N. Prec=−1, N. Match=PredicateMatch; N_1. Schema=该弧的Schema实现; N_1. Status=该弧Schema实现的ATN起始状态; N_1. Proposition=选中的Proposition的信息; N_1. Prec=C, N_1. Match=SchemaMatch.
被匹配的ATN弧没有Schema实现	 其中N. Schema=C. Schema; N. Status=当前ATN匹配Proposition到达的新状态; N. Proposition=null, N. Prec=−1, N. Match=PredicateMatch; N_i. Schema=C. Schema, N_i. Status=C. Status; N_i. Proposition=选中的各Proposition的信息; N_i. Prec=C, N_i. Match=PredicateMatch. $i=1,2,\ldots,p$	 其中N. Schema=C. Schema; N. Status=当前ATN匹配Proposition到达的新状态; N. Proposition=选中的Proposition的信息; N. Prec=−1, N. Match=PredicateMatch.

② 根据从当前结点出发具有下一优先权的弧的 Predicate 实现,选择与之匹配的 Proposition;

③ 如果下一个待匹配的弧为 jump 类型,按表 4 扩展焦点控制树.置 $C=N$. 然后转⑨;

④ 如果存在符合规则 1 的 Proposition,按照算法 1 处理,然后转⑨;

⑤ 如果存在符合规则 2 的 Proposition,按照算法 2 处理,然后转⑨;

⑥ 进行第 1 阶段回溯:

Table 3 Processing for controlling tree of focus in algorithm 3

表3 算法3对焦点控制树的处理

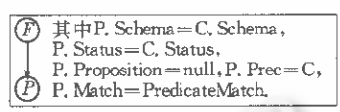
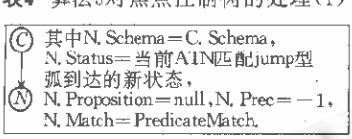


Table 4 Processing (1) for controlling tree of focus in algorithm 5

表4 算法5对焦点控制树的处理(1)



while 允许继续向上回溯(即不回溯到上一层次的 Schema)

{if 存在符合规则 3 的 Proposition,按照算法 3 处理,然后转⑨

当前结点=当前结点的父结点}

⑦ 如果存在符合规则 4 的 Proposition,按照算法 4 处理,然后转⑨;

⑧ 进行第 2 阶段回溯:

恢复第 1 阶段回溯前的当前结点 C,

while true

{删去当前结点的子树

if 从当前结点出发有更低优先权的弧,选择这样的弧进行匹配,然后转⑨

if 当前结点 C=当前结点 C 的 Schema 所对应子树的根结点

{if C. Match=SchemaMatch

{C. Match=PredicateMatch, temp=C. Prec,

C. Schema=temp. Schema, C. Status=temp. Status.

if 当前结点 C 有右子树 当前结点 C=当前结点 C 的右子树

else 当前结点 C=当前结点 C 的前驱结点的左子树,然后转⑨}

else 整个 Schema 匹配失败,算法结束}

当前结点 C=当前结点 C 的前驱结点〉

⑨ while 当前结点 C.Status 是 ATN 的终止状态

{当前结点 C=当前结点 C 的 Schema 所对应子树的根结点

if 当前结点 C 是整棵二叉树的根结点　整个 Schema 填充匹配成功,算法结束

else if 当前结点有右子树　当前结点 C=当前结点 C 的右子树

else 当前结点 C=当前结点 C 的前驱结点的左子树}

⑩ 转②.

注:为当前结点 C 增加右子树时,如果 C 的右子树已经非空,则先为 C 增加一个左子树,如表 5 所示,然后置 $C=P$.

Table 5 Processing (2) for controlling tree of focus in algorithm 5

表5 算法5对焦点控制树的处理(2)

C 其中P.Schema=C.Schema, P.Status=C.Status, P.Proposition=null, P.Prec=−1, P P.Match=PredicateMatch.

此外还需注意,在第 2 阶段回溯沿 Prec 链查找时,如果 C.Prec=−1,那么 C 的前驱结点就是其父结点.否则 C 的前驱结点是(C.Prec).Prec,即要向前回退两次.如果匹配得到多个 Proposition 符合同一焦点移动规则,若被匹配弧的重复特性为"+",则将这些 Proposition 按照指定的顺序排列后依次放置到焦点控制树上.若被匹配弧的重复特性为"1",则将这些 Proposition 按照指定的顺序排列后选择最前面的 Proposition 放置到焦点控制树上.

4　实验系统设计

根据改进的焦点控制算法设计了一个实验系统以验证算法的效果.实验系统主要是实现文本规划阶段的焦点控制及其与 Schema 填充的相关部分.系统的输入数据有:① 参数定义,定义 Predicate 中使用的各种参数的信息;② Predicate 定义:定义 Schema 中使用的 Predicate 的名称、参数格式等信息;③ Schema 定义:定义待填充匹配的 Schema 的结构;④ 知识库定义:定义填充匹配 Schema 的知识.所有输入信息都采用文本文件方式定义.为了便于实现,还对部分定义作了简化.实验系统根据上述定义来输入,利用焦点控制算法,输出填充匹配后的 Proposition 序列,并跟踪算法的中间过程.实验系统的总体结构如图 2 所示.

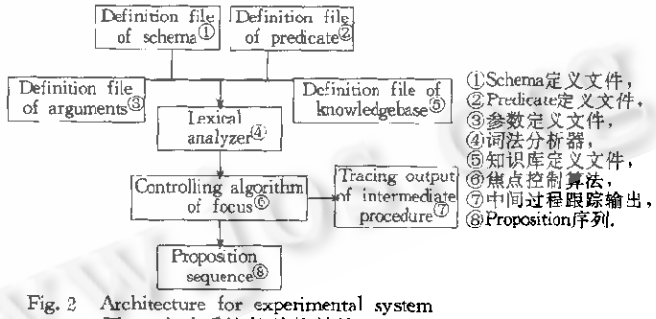


Fig. 2　Architecture for experimental system

图2　实验系统的总体结构

焦点控制算法在上一节已经作了详细描述,这里给出由焦点控制树得出 Proposition 序列的递归算法.

GetText(参数:焦点控制树的根结点 root)

(1) if root=−1,算法结束.

(2) if root 不是某一 Schema 所对应子树的根结点

{if root 的 Proposition 信息不为 null,输出 root 的 Proposition

GetText(root 的右子树根结点),

GetText(root 的左子树根结点)}

else {GetText(root 的左子树的根结点),GetText(root 的右子树根结点)}

下面,我们讨论实现中的若干问题.

(1) 可交换符号"‖"及其实现

Schema 表示中的可交换符号[4]的作用是使前后相关的若干成分的次序可以交换,文本生成时根据特定的条件进行排列. $P1 \| P2$ 相当于 $(P1P2)/(P2P1)$,对应的 ATN 如图 3 所示. 图中弧〈1,2〉和〈1,3〉的优先级是相同的. $P1$ 和 $P2$ 可能是由另一个 Schema 递归嵌套来实现,在状态 1 处判别 $P1$ 和 $P2$ 的取舍(即选择哪一条路径),实际上是取舍 $P1$ 和 $P2$ 的 Predicate 实现. 为此,在知识库中为每个 Proposition 设立优先级.

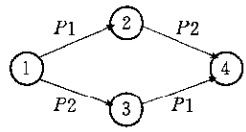


Fig. 3 ATN of $P1 \| P2$
图3 $P1 \| P2$所对应的ATN

Table 6 Schema of weather forecast
表6 天气预报的Schema

Title
WeatherAmplification*
StormAmplification*

当存在多个 Predicate 匹配得出的 Proposition 符合同一焦点移动规则时,选择优先级较高的一个. 以表 6 所示的天气预报 Schema 为例,在匹配了 Title 之后,分别匹配 WeatherAmplification 和 StormAmplification 的 Predicate 实现 Title 和 Attributive,得 Title:(1) 今天夜里和明天上海市和长江口区天气预报;Attributive:(2) 今年第 7 号台风的中心位置已经到达台湾省恒春市东南大约 620 公里的洋面上. 在夏季,可以设(2)的优先级>(1)的优先级,先填充台风的描述之一 StormAmplification,后填充天气的描述 WeatherAmplification;在其他季节,可以设(1)的优先级>(2)的优先级,先填充天气的描述 WeatherAmplification,后填充台风的描述 StormAmplification,这样可以突出两者在不同季节的重要性.

(2) 在不同的应用领域需要设置不同的辅助程序来确定 IRF 关系、Proposition 优先级和排序信息等内容. 这些信息的提供与各自的领域知识相关.

(3) IF 移动规则 1 的使用应注意避免出现焦点循环,即在下述序列中:

Proposition 1: CF=A PFL={B};

Proposition 2: CF=B PFL={A};

Proposition 3: CF=A PFL={...}.

Proposition 3 与前文的关系体现的应该是规则 3,而不是规则 1. 也就是说,在实际使用规则 1 时,PFL 除了要删去已被选中的 CF 外,还应该删去当时在焦点栈中的元素. CF 要转入焦点栈中的元素,即使该元素还出现在 PFL 中,也必须使用规则 3.

(4) 如果使用 IF 移动规则 1 或者规则 4 的弧的重复特性为"+"且该弧的 Schema 实现不是 null,那么每个选出的 Proposition 都可能扩展成另一个 Schema 所对应的描述. 此时应该对这些 Proposition 执行压缩操作,对焦点相同的 Proposition 只保留一个,避免多个展开的 Schema 描述具有相同的 GF.

5 结束语

新算法在以下几方面进行了改进:

(1) 改进了 Schema 的 ATN 表示. 使 Schema 表达式中带"+"和"*"的 Predicate 在 ATN 中有了两种表示方法,可以根据不同的文本特点选择相应的表示方法;

(2) 将栈式控制结构改进为树型控制结构,保留了所有填充过程中得到的 Proposition 及其焦点信息,实现了 Schema 的递归和填充匹配过程中的回溯,确定了 Schema 递归时 GF 的移动和控制方法. 为了实现回溯还在 ATN 的弧中增加了优先级;

(3) 完善了 IF 移动规则. 将 4 条规则同时运用,相对增加了文本焦点转移的灵活性,使算法适合更广泛的应用领域;

(4) 增加了对并列关系或句式相同的句子的排序,这是文本连贯性的一个重要方面. 虽然本文没有给出具体的排序方法(这与应用领域有关),但在实验系统中已加入了对排序结果的应用;

(5) 增加了对可交换符号的处理,为 Proposition 设立了优先级.

语言是多层次的,因此语篇的衔接与连贯必然在不同层面上有所反映,包括音系层、句法层、词汇层、语义层、社会符号层等[6]. 本文的算法主要是针对语义层使生成的文本保证连贯性,这与具体的自然语言语种是无关的. 文本生成过程中还可以在句法层(包括句式选择、结构衔接等)和词汇层(包括指称、词汇衔接等)等层次上研

究文本连贯性控制的方法.这就与具体的语种相关了.

参考文献

1 Cole R A *et al*. Survey of the state of the art in human language technology. http://www.coli.uni-sb.de/~hansu. The Web Page for the Department of Computational Linguistics at the University of the Saarland. 1996

2 Mckeown K R. Text Generation: Using Discourse Strategies and Focus Constraints to Generate Natural Language Text. Cambridge: Cambridge University Press, 1985. 30～35, 57～58, 71～73

3 Yao Tian-shun. Natural Language Understanding: A Study of Making a Machine Understand Human Language. Beijing: Tsinghua University Press, 1995. 54～55
(姚天顺.自然语言理解——一种让机器懂得人类语言的研究.北京:清华大学出版社,1995.54～55)

4 Wang Xian, Yao Tian-fang. Design and implementation of macroplanner for Chinese weather forecast automated generation system. In: Chen Li-wei, Yuan Qi eds. Language Engineering. Beijing: Tsinghua University Press, 1997. 176～181
(王纤,姚天昉.汉语天气预报文本内容规划器的设计与实现.见:陈力为,袁琦编.语言工程.北京:清华大学出版社,1997.176～181)

5 Cai Zi-xing, Xu Guang-you. Artificial Intelligence: Principles and Applications. 2nd ed. Beijing: Tsinghua University Press, 1996. 307～309
(蔡自兴,徐光祐.人工智能及其应用.第2版.北京:清华大学出版社,1996.307～309)

6 Hu Zhuang-lin. The Joining and Coherence of Discourse. Shanghai: Shanghai Foreign Language Education Press, 1994. 224～226
(胡壮麟.语篇的衔接与连贯.上海:上海外语教育出版社,1994.224～226)

Research on the Shift Controlling Algorithm of Global and Immediate Focuses in Text Planning

YAO Tian-fang TANG Xue-yan

(*Department of Computer Science and Engineering Shanghai Jiaotong University Shanghai* 200030)

Abstract The improvement on the shift rules of global and immediate focuses is presented. The improved rules can be used for more extensive domain. Based on Mckeown's focus algorithm, a new algorithm that adopts tree as a controlling structure instead of stack is proposed. It can control the shift of both global and immediate focuses simultaneously. The new algorithm improves the ATN representation of Schema by adding repeat property and priority information to the edge and incorporates the control of the recursion of Schema and backtrack mechanism. The filling and recursion of Schema are determined by the knowledge base. Moreover, the new algorithm can also handle the interchangeable symbols in the Schema. The design of an experimental system based on the proposed algorithm is introduced and the effectiveness of the algorithm is discussed.

Key words Text planning, global focus, immediate focus, Schema, predicate.