

一种基于广义随机 Petri 网的 BPEL 建模

谭文安^{1,2}, 葛铭珏¹

(1. 南京航空航天大学计算机科学与技术学院, 江苏 南京 210016)
(2. 上海第二工业大学计算机与信息学院, 上海 201209)

[摘要] BPEL 是一种广泛应用于支撑跨组织协同服务的组合标准. 为分析 BPEL 语言构造业务过程的性能, 基于广义随机 Petri 网, 提出了一种 BPEL 模型, 详细地刻画了 BPEL 的语法语义、活动的不同状态和执行机制, 案例分析说明该模型可以有效支持服务组合定量分析. 最后对未来工作进行了展望.

[关键词] BPEL, GSPN, 服务组合, 建模

[中图分类号] TP311 [文献标志码] A [文章编号] 1001-4616(2014)01-0017-05

A General Stochastic Petri Net-Based Representation for Modeling BPEL

Tan Wenan^{1,2}, Ge Mingjue¹

(1. School of Computer Science and Technology, Nanjing University of Aeronautics and Astronautics, Nanjing 210016, China)
(2. School of Computer and Information, Shanghai Second Polytechnic University, Shanghai 201209, China)

Abstract: Business Process Execution Language (BPEL) is a widely utilized web service composition standard to support cross-organizational collaboration. For analyzing the performance of the process generated through BPEL, we propose a way to construct more reliable cross-workflow applications. Basic on General Stochastic Petri Net (GSPN), a BPEL model is proposed in this paper. A detailed description of the BPEL syntax, semantic, activity's statues and the execution mechanism and logic are introduced. Finally, a practical case study has proved this representation suitable for support performance analysis of the proposed model quantitatively and the future works are discussed.

Key words: BPEL, GSPN, service composition, modeling

随着面向服务的架构 SOA (Service Oriented Architecture) 技术不断的应用, 计算机支持的跨组织协同工作 CSCW (Computer Supported Cooperative Work) 得到了极大的发展. 其中, 服务组合技术被认为是跨企业资源重组的重要手段. 2003 年首次提出的业务过程执行语言 BPEL (Business Process Execution Language) 是一种广泛应用的服务组合标准. 目前 BPEL 语言不能对其构造的服务组合性能指标进行评估和描述. 随着服务组合工作流可靠性需求的持续增长, 研究基于 BPEL 的建模和分析方法对服务组合性能进行分析是十分有意义.

Petri 网是一种严谨的直观图形化建模方法. Ouyang^[1] 等人研究了 BPEL 控制流程和规范语义的分析. Yang Y P^[2] 等人研究了基于分层 Petri 网技术应用于服务组合. 他们的研究成果^[3,4] 催生了许多实际应用的分析工具^[5]. 传统的 Petri 网中不包含时间与概率, 只能支持服务组合的活性分析、有界性分析和是否包含死锁等定性分析, 未能满足过程定量分析的需求. 因此, 补充时间和概率的广义随机 Petri 网 (General Stochastic Petri Net, GSPN) 成为传统 Petri 网建模的有效补充^[6,7]. Xia Y^[8] 等人设计了由 BPEL 向 GSPN 模型的映射方法, 并提出了一个基于状态的性能分析方法, 其设计模型考虑了 BPEL 语言结构化特点, 具有建设性, 但缺少对活动状态和 BPEL 流程执行机制的刻画, 因而无法投入实际的应用. 因此, 针对研究现状, 提出了一种基于广义随机 Petri 网的 BPEL 建模方法. 该模型详细地刻画了活动的状态并精确地描述了 BPEL 的执行过程, 能有效地支持服务组合过程的定量分析.

收稿日期: 2013-08-10.

基金项目: 国家自然科学基金 (61272036)、上海第二工业大学软件工程重点学科 (XXKZD1301).

通讯联系人: 谭文安, 博士, 教授, 研究方向: 软件工程、协同计算、智能信息系统. E-mail: twajs@sohu.com

1 BPEL 的建模

1.1 广义随机 Petri 网

广义随机 Petri 网是传统 Petri 网的扩展,引入了变迁发生的时延以及变迁发生的概率.

定义 1 广义随机 Petri 网(GSPN)是一个五元组 $(P, T, F, M_0 | \Lambda)$, 其中:

- (1) $P = \{p_1, p_2, \dots, p_k\}$ 是所有库所的集合,库所表示系统的不同状态;
- (2) $T = \{t_1, t_2, \dots, t_m\}$ 是所有变迁的集合,变迁表示系统状态间的变化过程. $T = T_M \cup T_1$, 表示变迁分为时间变迁 $T_M = \{t_1, t_2, \dots, t_n\}$ 和瞬时变迁 $T_1 = \{t_{n+1}, t_{n+2}, \dots, t_m\}$;
- (3) $F \subseteq (P \times T) \cup (T \times P)$ 是所有弧的集合,弧表示系统状态和变化过程的对应关系;
- (4) $M_0 = (m_{01}, m_{02}, \dots, m_{0k})$ 是初始状态,表示所有库所在初识时的令牌数;
- (5) $\Lambda = \{\lambda_1, \lambda_2, \dots, \lambda_n\}$ 是对应时间变迁的平均变迁发射速率的集合. 时间变迁的发射满足指数分布,平均实施速率的倒数 $\tau_i = 1/\lambda_i$ 为变迁 t_i 的平均实施延时或平均服务时间.

1.2 活动状态

活动是 BPEL 语法中最基本的元素,分为基本活动(Basic Activity)、结构化活动(Structure Activity)和特殊的<scope>活动. BPEL 以活动的状态为基础,对活动及活动间相互作用进行控制,如图 1(左)所示,其中 N 作为活动名称的占位符,并用虚线矩形框表示活动,活动名称标示在框中. 本文任何模型都包括:开始状态 p_{N-Init} , 完成状态 p_{N-Com} , 故障抛出状态 $p_{N-Fault}$ 以及活动终止状态 p_{N-Stop} 与已终止状态 $p_{N-stopped}$.

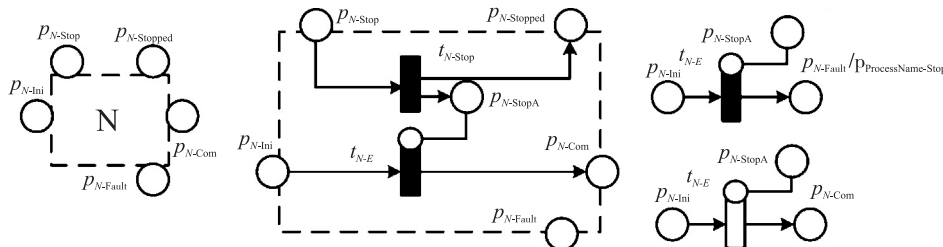


图 1 活动表示(左),基本活动的建模(中),<throw>,<exit>和<wait>的说明(右)

Fig. 1 Expression of activity, model for basic activity, special caption for <throw>, <exit> and <wait>

1.3 基本活动的建模

基本活动是服务组合流程的基本执行单元. BPEL 设计规范了多种不同类型的基本活动以完成不同的功能. 如图 1(中)的建模所示,多数基本活动对服务组合流程性能鲜有影响而未考虑其出现故障. 部分特殊功能的活动需要在此模型基础上适当调整:活动<throw>抛出指定的故障信息,应如图 1(右)将 t_{N-E} 发出的弧指向 $p_{N-Fault}$;活动<wait>的模型中,将 t_{N-E} 改为时间变迁以使其等待时间;活动<exit>使整个流程终止,故使 t_{N-E} 发出的弧指向 $p_{Process Name-Stop}$ (流程的终止状态).

活动<invoke>是所有基本活动中唯一与伙伴服务产生交互的活动,因此对服务组合性能的表现有重要影响. <invoke>向远程伙伴服务提供者发送 SOAP(Simple Object Access Protocol)消息建立服务调用连接,这个连接可能因伙伴服务失效或消息的丢失造成活动的故障. 图 2(左)中分别用 t_{N-SF} 、 t_{N-SW} 和 t_{N-SO} 表示伙伴服务失效、消息丢失及建立服务调用连接三种情况. 时间变迁 t_{N-SWD} 表示等待已丢失消息的耗时,变

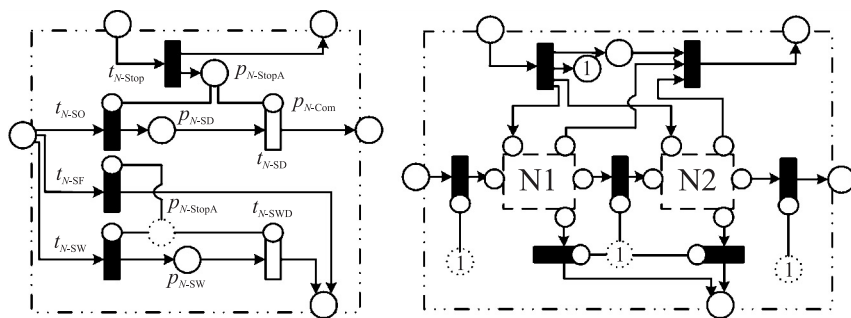


图 2 <invoke>的建模(左)和<sequence>的建模(右)

Fig. 2 Model for <invoke> and model for <sequence>

迁 t_{N-SD} 表示伙伴服务运行的耗时. $\langle \text{invoke} \rangle$ 活动也可包含补偿处理及故障处理模块,语义等同于 $\langle \text{scope} \rangle$ 活动,其讨论见 1.5 节.

1.4 结构化活动的建模

结构化活动规定活动间执行次序及控制模式. 活动 $\langle \text{sequence} \rangle$ 是顺序执行活动结构. 如图 2(右)所示,活动 $N1$ 执行后才激发活动 $N2$,其中 $p_{N-StopA}$ (标号 1) 发出的抑制弧在活动终止时抑制这些变迁的发生.

如图 3(左)所示,活动 $\langle \text{flow} \rangle$ 规定包含的活动并发执行,其逻辑控制机制讨论见 1.6 节. 活动 $\langle \text{if} \rangle$ 和 $\langle \text{pick} \rangle$ 的结构有所不同. $\langle \text{if} \rangle$ 活动开始后顺序检测各分支满足条件,当满足时执行包含的活动,如图 3(中)用 t_{N-IF} 和 t_{N-Else} 表示不同分支;图 3(右)模型 $\langle \text{pick} \rangle$ 活动中每条分支对应不同事件的响应,包含了响应消息事件的变迁 t_{N-PT} 和响应时钟事件的变迁 t_{N-PF} ,用时间变迁 t_{N-PFW} 表示时钟事件的耗时.

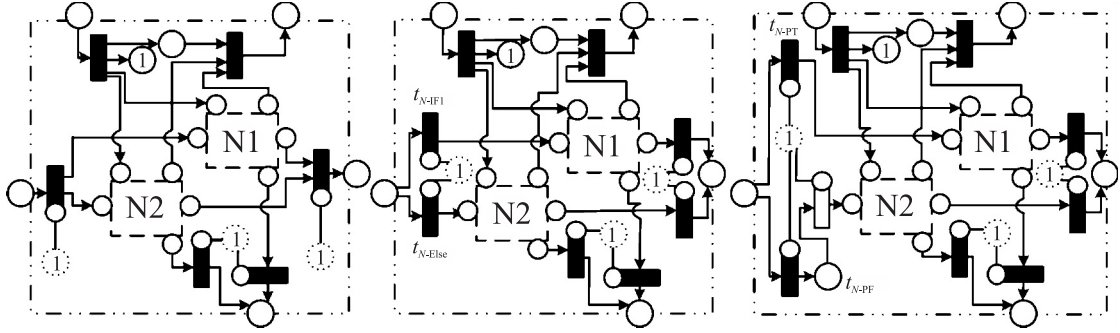


图 3 $\langle \text{flow} \rangle$ 的建模(左), $\langle \text{if} \rangle$ 的建模(中)和 $\langle \text{pick} \rangle$ 的建模(右)

Fig. 3 Model for $\langle \text{flow} \rangle$, model for $\langle \text{if} \rangle$ and model for $\langle \text{pick} \rangle$

活动 $\langle \text{while} \rangle$ 和 $\langle \text{repeatUntil} \rangle$ 循环结构如图 4(左,中)所示,其中变迁 t_{N-CT} 表示满足循环条件. $\langle \text{forEach} \rangle$ 活动也提供循环结构,但仅包含 $\langle \text{scope} \rangle$ 活动并循环固定的 $m+1$ 次,其中 $m = \langle \text{finalCounterValue} \rangle - \langle \text{startCounterValues} \rangle$. 图 4(右)中,库所 p_{N-RC} (标号 2) 在活动初始时获得 m 令牌以限定循环的次数.

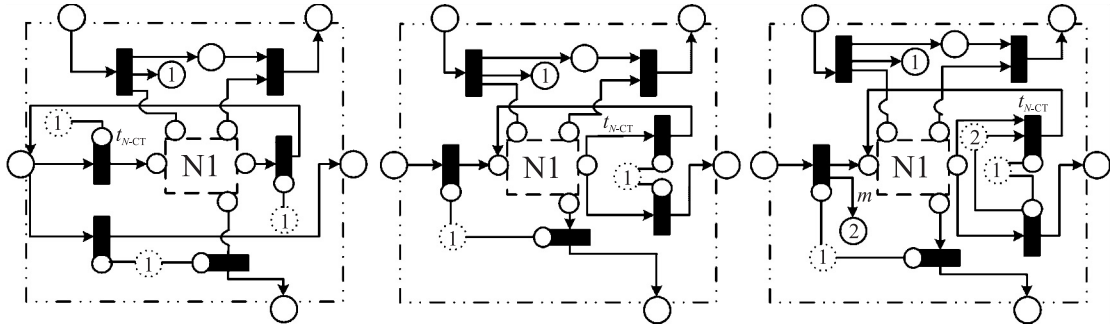


图 4 $\langle \text{while} \rangle$ 的建模(左), $\langle \text{repeatUntil} \rangle$ 的建模(中)和 $\langle \text{forEach} \rangle$ 的建模(右)

Fig. 4 Model for $\langle \text{while} \rangle$, model for $\langle \text{repeatUntil} \rangle$ and model for $\langle \text{forEach} \rangle$

1.5 $\langle \text{scope} \rangle$ 活动的建模

活动 $\langle \text{scope} \rangle$ 为内嵌主活动 MA 提供行为控制:故障处理 FH、事件处理 EH、补偿处理 CH 和终止处理 TH. FH 捕获主活动抛出的故障信息 EH 并发执行处理不同事件,主活动结束后,CH 可以进行其额外的处理,当 $\langle \text{scope} \rangle$ 活动被终止时触发 TH. 图 5(左)中 $p_{N-CTrigger}$ 和 $p_{N-Etrigger}$ (标号 1 和 2) 捕获相应的触发事件进行相应的处理活动, $p_{N-Clinhibitor}$ (标号 3) 表示故障处理发生后禁止补偿处理, $p_{N-Elinhibitor}$ (标号 4) 表示故障处理开始即禁止触发新的事件处理, $p_{N-Ecount}$ (标号 8) 及其抑制弧表示 $\langle \text{scope} \rangle$ 活动必须等待所有事件处理完成, $p_{N-EAbsorb}$ (标号 9) 用于吸收 $p_{N-Ecount}$ 中的令牌.

1.6 $\langle \text{link} \rangle$ 和 DPE

活动 $\langle \text{flow} \rangle$ 中使用 link 及 DPE (dead-path-elimination, 死链删除机制) 实现扩展控制依赖关系. link 表示含有 $\langle \text{source} \rangle$ 源活动与含有相同名称 $\langle \text{target} \rangle$ 目标活动间的对应关系. 源活动根据 $\langle \text{transitionCondition} \rangle$ 值为 link 赋值 (true/false). 当所有 link 已赋值,且 $\langle \text{JoinCondition} \rangle$ 结果为真,目标活动顺利执行;若为假,且 $\langle \text{suppressJoinFailure} \rangle$ 为真时,跳过目标活动;否则抛出 JoinFailure 故障消息. 图 5(右)展示了与 link 有

关的活动建模. 其中虚线矩形 Sub 表示未考虑 link 的原活动建模, 并将名称开头改为 N-Sub. 虚线矩形“J”将 $\langle \text{JoinCondition} \rangle$ 中布尔表达式映射为 Petri 网^[9]. p_{N-I} 和 p_{N-IF} (标号 5, 6) 分别表示 $\langle \text{JoinCondition} \rangle$ 不同的结果, p_{N-Spi} 捕获 $\langle \text{suppressJoinFailure} \rangle$ 的值, 标号为 3 和 4 的库所分别表示 link 在赋值后的 true 或 false 值.

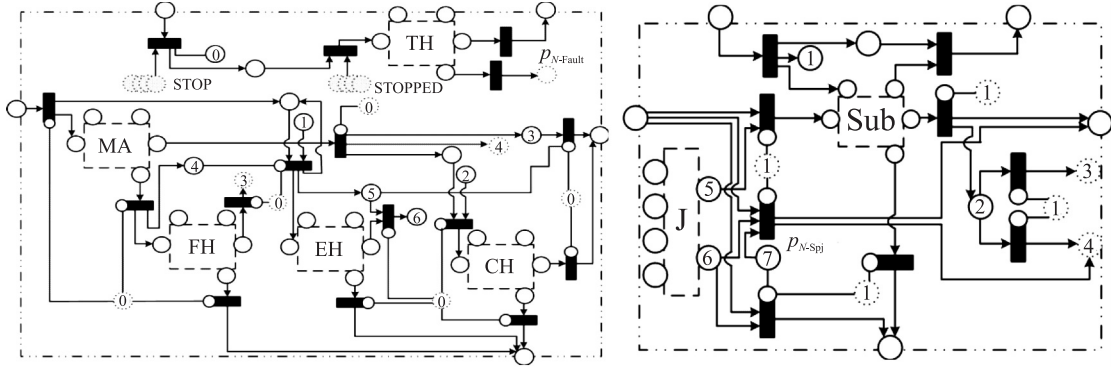


图 5 $\langle \text{scope} \rangle$ 活动的建模(左)和 link 相关活动的建模(右)

Fig. 5 Model for $\langle \text{scope} \rangle$ and model for activity related with link semantic

2 实例分析与相关工作

图 6 左侧展示了该代码片段对应的 GSPN 模型, 其中部分抑制弧未画出. 在建模完成后, 通过不同的分析方法可以对服务组合流程的性能进行评估. 例如文献[8]中提出基于状态的期望结束时间(ENPCT, Expected Process Normal Completion Time)分析方法. 在图 6 中, 我们将 p_{F-Com} 视为正常完成状态, 而 $p_{F-Fault}$ 作为未完成状态. 那么, $EDT(i)$ 计算了由状态标记 m_i 到正常完成状态标记(即 p_{F-Com} 中拥有令牌的状态标记)的期望耗时. 因此流程的 ENPCT 等于初始状态标记 m_0 的 $EDT(0)$.

$$EDT(i) = \begin{cases} 0, & \text{若 } m_i \text{ 是正常完成状态标记,} \\ \infty, & \text{若 } m_i \text{ 是未完成状态标记,} \\ \infty, & \text{若 } \forall s(j) \in IMS(i), EDT(j) = \infty, \\ -\frac{n}{q_{i,i}} + \sum_{1 \leq k \leq |S|, k \neq i, EDT(k) < \infty} \frac{q_{i,k} \times EDT(k)}{TP(i)}, & \text{其他情况,} \end{cases}$$

其中 $TP(i) = \sum_{1 \leq k \leq |S|, k \neq i, EDT(k) < \infty} q_{i,k} = -q_{i,i}$.

若在模型中, 设定活动 A 中三个变迁的随机开关为: $pe(t_{A-sub-SO}) = 0.91$, $pe(t_{A-sub-SF}) = 0.03$ 和 $pe(t_{A-sub-SW}) = 0.06$. 若当变量 \$guest.hurry 不为 No 的概率为 0.31, 即 $pe(t_{AtoE-F}) = 0.31$ 和 $pe(t_{AtoB-T}) = 0.69$, 且设定时间变迁的平均发射概率 $\lambda(t_{A-sub-SD}) = 0.032$ (平均延时约 31 s), $\lambda(t_{A-sub-SWD}) = 0.125$ (平均等待约 8 s) 和 $\lambda(t_{B-sub-E}) = 0.058$

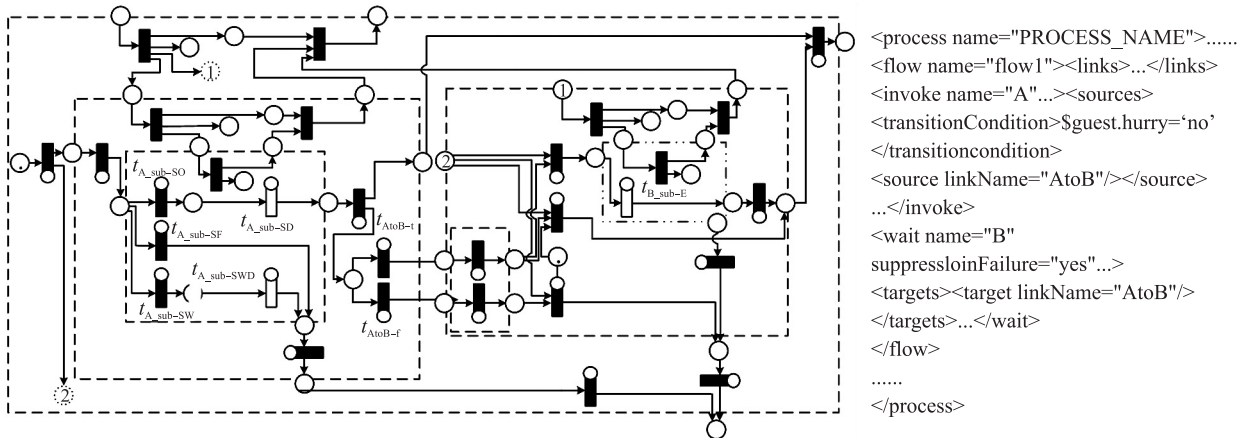


图 6 BPEL 代码段及建模实例

Fig. 6 BPEL segment and its GSPN model

(平均延时约 17 s),通过以上公式得出 $ENPCT=EDT(0)\approx 39.26$ s,即期望结束时间为 39.26 s.

本文模型更精确地讨论了 GSPN 描述的服务组合流程以及执行过程,较完备地涵盖了不同的活动类型.虽然与文献[8]相比增加了部分库所及变迁,但是并未极具扩张状态空间,在使用相似效率执行分析方法时更为精确.与文献[5]的语法模型相比较,本文模块化的建模支持了多层面的性能分析,有效地支持自动化的建模工作.

3 总结

本文提出了一种基于广义随机 Petri 网的 BPEL 建模方法,详细地刻画了活动的状态及 BPEL 的执行过程,有效地支持了服务组合流程的定量分析.并且本文对服务组合流程进行模块化、结构化建模,提供了多层面的性能分析,有效地支持自动建模的实现,为今后的工作奠定了基础.未来的工作将围绕模型化简,模型参数的设定及定量分析算法的优化等展开.

[参考文献]

- [1] Ouyang C, Verbeek E, van der Aalst W M P, et al. Formal semantics and analysis of control flow in WS-BPEL[J]. Science of Computer Programming, 2007, 67(2/3): 162–198.
- [2] Yang Y P, Tan Q P, Xiao Y. Verifying web services composition based on hierarchical colored Petri nets[C]//Proceedings of the First International Workshop on Interoperability of Heterogeneous Information Systems. New York: ACM, 2005: 47–54.
- [3] 王玉英, 陈平. 基于有色 Petri 网的 BPEL 中 flow 活动建模[J]. 计算机应用研究, 2011(28): 632–635.
- [4] Liu X, Li J H, Li X, et al. BPEL modeling based on extended synchronized Petri net[J]. 计算机工程, 2011(37): 57–59.
- [5] Niels Lohmann. A feature-complete Petri net semantics for WS-BPEL 2.0[J]. Lecture Notes in Computer Science, 2008(4 937): 77–91.
- [6] Henrique Jorge A Holanda, Barroso G C, Serra A, et al. SPEWS: a framework for the performance analysis of web services orchestrated with BPEL4WS[C]//Fourth International Conference on Internet and Web Applications and Services. Venice, 2009: 363–369.
- [7] Wang W P, Gao Z L, Bao S Y, et al. The development of reliability modeling and analysis tool based on stochastic Petri nets[J]. Advanced Materials Research, 2010(118/120): 566–570.
- [8] Xia Y, Liu Y, Lin J, et al. Modeling and performance evaluation of BPEL processes: A stochastic-Petri-net-based approach[J]. IEEE Transactions on Systems, Man and Cybernetics—Part A: Systems and Humans, 2012(42): 503–510.
- [9] Pastor E, Roig O, Cortadella J, et al. Petri net analysis using Boolean manipulation in Proc Appl[J]. Theory Petri Nets, 1994(1): 416–435.

[责任编辑: 丁 蓉]