ISSN 1000-9825, CODEN RUXUEW
Journal of Software, Vol.20, No.3, March 2009, pp.557−566
doi: 10.3724/SP.J.1001.2009.03314

E-mail: jos@iscas.ac.cn
http://www.jos.org.cn
Tel/Fax: +86-10-62562563

一种用于软件过程建模的适应性 Agent 协商*

黎 巎[1,2], 李明树[1,3+], 王 青[1], 赵 琛[1], 杜栓柱[1]

[1](中国科学院 软件研究所 互联网软件技术实验室,北京 100190)

[2](中国科学院 研究生院,北京 100049)

[3](中国科学院 软件研究所 计算机科学国家重点实验室,北京 100190)

Adaptive Agent Negotiation for Software Process Modeling

LI Nao[1,2], LI Ming-Shu[1,3+], WANG Qing[1], ZHAO Chen[1], DU Shuan-Zhu[1]

[1](Laboratory for Internet Software Technologies, Institute of Software, The Chinese Academy of Sciences, Beijing 100190, China)

[2](Graduate University, The Chinese Academy of Sciences, Beijing 100049, China)

[3](State Key Laboratory for Computer Science, Institute of Software, The Chinese Academy of Sciences, Beijing 100190, China)

+ Corresponding author: E-mail: mingshu@iscas.ac.cn

Li N, Li MS, Wang Q, Zhao C, Du SZ. Adaptive agent negotiation for software process modeling. *Journal of Software*, 2009,20(3):557−566. http://www.jos.org.cn/1000-9825/3314.htm

Abstract: Most Software process models are predefined. When applied in changing environments, they have to be adapted manually. To this end, this paper proposes an adaptive multilateral negotiation model for software process modeling, namely AMNM-PA. AMNM-PA uses Agents to represent the entities involved in software processes, such as organizations, teams, persons, etc. and dynamically and adaptively constructs software process models for given software projects by negotiating among the Agents. AMNM-PA is based on non-stationary finite-horizon Markov decision processes and uses the model-independent Q learning algorithm to choose negotiation strategies, thus supports the dynamic and adaptable negotiation in changing and unknown environments meeting the requirement for environmental adaptability of the software process modeling. AMNM-PA has been implemented in the software process management system SoftPM.

Key words: negotiation model; adaptability; Q-learning; software process modeling; agent

摘 要: 大多软件过程模型是预定义的.在变化的应用环境中,需要由相应人员进行适应性调整.提出一种用于软件过程建模的适应性多边协商模型——AMNM-PA,其采用 Agent 封装软件过程中所涉及的个体,包含组织、团队、个人等,通过 Agent 间的协商动态、适应地建立针对给定软件项目的软件过程模型.AMNM-PA 基于非静态有限阶段 Markov 决策过程,采用模型无关的 Q 学习算法选取协商策略,因此能够支持动态、非预知环境下的适应性协商,

* Supported by the National Natural Science Foundation of China under Grant Nos.60573082, 60673121, 90718042 (国家自然科学基金); the National High-Tech Research and Development Plan of China under Grant Nos.2006AA01Z185, 2006AA01Z19B, 2007AA010303 (国家高技术研究发展计划(863)); the National Key Technologies R&D Program of China under Grant No.2005BA113A01 (国家科技攻关计划); the National Basic Research Program of China under Grant No.2007CB310802 (国家重点基础研究发展计划(973))

Received 2008-01-07; Accepted 2008-02-27

从而满足软件过程建模对环境的适应性需求.AMNM-PA 已经实施于软件过程管理系统——SoftPM.

关键词: 协商模型;适应性;Q 学习;软件过程建模;Agent

中图法分类号: TP311 **文献标识码:** A

软件过程不同于一般意义的工业过程或流程较为稳定的业务过程,要求更高的灵活性和适应性.软件过程参与者的及时反应能力是软件过程能够灵活适应各种变化的关键因素.目前,个体在软件过程中的重要性已经逐渐得到认识和重视,如 PSP,TSP 的提出、发展和应用.然而,对于软件过程建模而言,传统方法和工具无法支持这种人为主体、高度灵活的过程模式[1,2].

Agent 的提出源于分布式问题求解,因此软件过程建模多采用 Agent 技术解决分布式、非集中环境的执行自动与过程协同[3−6].这些方法利用 Agent 的自动、交互特性,为软件过程执行提供了一定的灵活性.然而,个体在软件过程中的作用并未得到足够重视.目前,由于 Agent 的特性——自治、自主、智能、协作等能够很好地刻画人在软件过程中所表现的行为,因此,Agent 技术在软件过程领域的应用已经成为软件过程领域的重要研究方向之一[7].

我们提出了一种基于 Agent 的软件过程建模方法(OEC-SPM)[8],并开发了相应的软件工程环境 SoftPM[9−11].相比其他采用 Agent 技术的过程建模方法,OEC-SPM 的主要特点是人为中心、合同驱动,由此建模和执行都具有很好的灵活性.具体地,OEC-SPM 用 Agent 封装软件过程中涉及的个体,即个人及个人组成的团队、部门、组织等,包含个体的软件过程知识、对环境的感知能力与其他 Agent 的协商能力等;这些 Agent 通过对环境进行感知,动态建立针对给定软件项目的软件过程模型,并在执行过程中不断更新自身知识和能力以动态适应环境的变化.

协商是 OEC-SPM 动态建立软件过程模型的方式.在 OEC-SPM 中,给定软件项目后,某管理者 Agent(如项目经理)对该项目进行首层任务划分,然后就每个划分的任务发起与其他 Agent 的一对多协商.协商主题即任务的属性,如工作量、质量、获益等.当某 Agent 与管理者 Agent 就某任务及任务属性值达成一致后,其可能对该任务进行下一层任务划分,由此发起其与其他 Agent 的一对多协商.任务划分与协商一直进行至再无任务划分时为止.由此建立的针对该项目的自上而下的 Agent 组织结构,包含了所有承担该项目任务及子任务的 Agent 及每个 Agent 就所承担的任务的向上和向下的承诺——彼此间的协作合同,称为合同网.合同网即为 OEC-SPM 的软件过程模型.软件过程对环境的依赖性决定了上述用于过程建模的协商需要具有环境适应性.本文所提到环境适应性是指当建模环境发生变化时,如协商剩余时间、参与协商的过程 Agent 数目、受众需求、市场状况等,参加协商的 Agent 能够进行决策和行为调整而适应这些变化.

适应性协商的研究主要集中于基于 Bayesian 学习和强化学习——特别是 Q 学习的协商决策中.Bazaar[12]是一个基于连续决策过程的协商模型,在每个决策点,Agent 根据更新的 Bayesian 信念选择适应性的动作.Bazaar 的 Bayesian 学习是静态环境下的学习.Q 学习是目前研究适应性协商的重要方法,且大多以 Markov 决策过程为架构,是一种模型无关的强化学习算法.Q 学习在无须预知环境模型的情况下,以环境-动作对的奖赏值为依据给出优化动作.基于经典 Markov 决策过程(MDP's)的 Q 学习仅有单一的 Agent 与环境进行交互,其他 Agent 被看成环境的一部分而不具有适应性,因此并不适用于所有 Agent 均需与环境进行交互的多 Agent 系统.文献[13,14]用 Markov 对策代替 MDP's,使参与协商的 Agent 均具有适应性,从而可以应用于多 Agent 系统.但其主要解决无限阶段的 Markov 决策问题,且环境定义仅依赖于对手信息,忽视了其他环境因素对协商的影响.文献[15]提出了一种非静态有限阶段 Markov 决策过程模型,能够支持动态环境下的有限阶段协商,但仍然存在决策方法仅依赖对手信息的缺点.文献[16]基于文献[13−15]的工作,针对电子商务环境下的环境因素及它们的动态变化性,提出了一种适应的、双边、单主题协商模型.由于其采用值迭代算法求解 Markov 决策问题而不是 Q 学习算法,因此需要预定义环境模型.一般而言,在动态变化的环境中,环境模型的定义是困难的.一个典型的例子是网格环境.文献[17]基于环境模型无关的 Q 学习算法的双边协商对动态的、无法预测的网格环境进行资源调度.

OEC-SPM 中的协商是双赢协商或称非零和对策,主要特点:一是协商决策侧重对环境状态而不是对手的分析,这表明相对于电子商务应用等其他领域,软件过程建模中的协商强调的是合作而不是竞争;二是环境的变化具有未知性,很难预定义或预测环境的变化;三是多边性,即如前所述的一对多协商.因此,上述方法在引入 OEC-SPM 时,存在不能完全适应的问题.我们于先前工作中所建立的用于 OEC-SPM 的协商模型侧重于多边协商策略[18].本文针对软件过程建模对环境的强依赖性,用非静态有限阶段 Markov 决策过程[15]对该模型进行了扩展,扩展后的模型称为 AMNM-PA.在 AMNM-PA 中,协商过程由多个非静态有限阶段 Markov 决策过程构成,即每个参与协商的 Agent 都拥有各自的非静态有限阶段 Markov 决策过程;在每个决策过程中,Agent 基于当前捕捉到的环境状态,采用模型无关的 Q 学习算法选取适应的协商策略,从而使协商行为、协商结果(即软件过程模型)具有环境适应性.

1 适应性多边协商模型

用于 OEC-SPM 即软件过程建模的适应性多边协商模型 AMNM-PA 主要包含 Markov 决策过程,具体地,非静态有限阶段 Markov 决策过程是架构的决策过程集和该集中每个决策过程必须遵循的协商协议.Markov 决策过程的基本性质是过程中的未来行动仅依赖于当前状态和当前状态下的行动而与之前的历史无关.由此,AMNM-PA 的一个基本设定是:某时刻的协商行动仅依赖于该时刻的前一时刻.在此前提下,定义 1 给出了 AMNM-PA 的形式定义.

定义 1. 适应性多边协商模型 AMNM-PA 是一个 4 元组(Pa,O,P,DP),其中,

(1) Pa 是参与协商的过程 Agent 集,$Pa=\{pa_1,pa_2,\ldots,pa_n\}$.过程 Agent 是指能够实施软件过程的 Agent,以下均简称 Agent. 这里区分两种 Agent,一为协商发起者,二为协商响应者(即除发起者之外参与协商的 Agent).

(2) O 是协商主题集,$O=\{o_1,o_2,o_n\}$.Agent 向其他 Agent 发送的协商主题值集构成一个 offer,offer 表示 Agent 对协商主题值的意愿.根据软件过程特性,最终达成一致的 offer 称为协作合同(CC),即一旦 Agent 就协商主题值达成了一致,这些协商主题值就构成了 Agent 间的协作约束.

(3) P 是协商协议,即每个 Markov 决策过程必须遵循的公共规则.P 是一个 5 元组($NS,NPrim,tf,of,Thr$).图 1 描述了 P 所包含的元素及其之间的关系,NPrimi→是协商发起者发送的协商原语,→NPrimi 是协商响应者发送的协商原语,NPrimi 是任意方发送的协商原语.

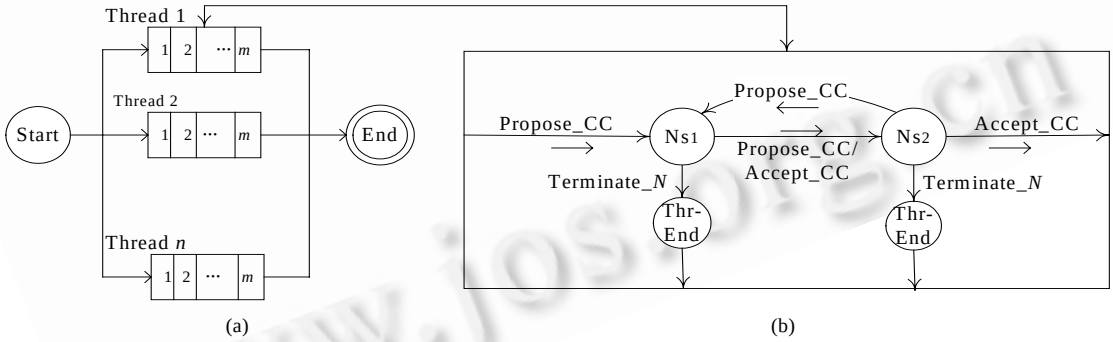

Fig.1 Multilateral negotiation protocol

图 1 多边协商协议

1) NS 是协商状态类别集,$NS=\{Start,Ns_1,Ns_2,Thr\text{-}End,End\}$,$Start$ 是协商初始状态,End 是协商结束状态,Ns_1 和 Ns_2 是协商中间状态,$Thr\text{-}End$ 是协商线程(见下面第 5))结束状态.

并非所有参与协商的 Agent 都经历所有协商状态类别.这是因为,在 offer 没有达成一致之前,某些 Agent 可能进入协商线程结束状态而退出协商.

2) $NPrim$ 是协商原语集,用于 Agent 间的话语通信,$NPrim=\{Propose\_CC,Accept\_CC,Terminate\_N\}$,其

中,*Propose_CC* 表示发送 offer(即发送意愿协作合同),*Accept_CC* 表示接受 offer(即接受对方的意愿协作合同),*Terminate_N* 表示终止协商线程.

3) *tf* 是协商状态类别转换函数,$tf:NS\times NPrim\to NS$,定义在每个协商状态类别发送和接收某原语后协商状态类别的转换.图 1(b)中的箭头表示了 *tf*.注意,除了一致达成协商结束以外,当所有协商响应者均到达协商线程结束状态时,协商也正常结束.

4) *of* 是协商状态类别的输出函数,$of:NS\times NPrim\to NPrim$,定义在每一协商状态类别所允许接收和发送的协商原语,如图 1(b)中箭头上的原语所示.

5) *Thr* 是协商线程集$\{Thr_1,Thr_2,\ldots,Thr_n\}$.$Thr_i$ 是单个线程,由协商发起者(由 *b* 表示)和协商响应者(由 *i* 表示,$i=1,\ldots,n$)交替发送的消息 *M* 序列构成,$Thri=\{M_{b\to i}^{t1}[1],M_{i\to b}^{t2}[1],M_{b\to i}^{t3}[2],M_{i\to b}^{t4}[2],\ldots\}$,其中,$M_{b\to i}^{tk}[m]$(或 $M_{i\to b}^{tk}[m]$)表示由 *b*(或 *i*)在"第 *m* 次交互"("一次交互"是指协商发起者发送消息给所有响应者并获得所有响应者的回应的过程;下一次交互从协商发起者发送下一个消息给所有响应者开始),在 *tk* 时刻向 *i*(或 *b*)发送的消息($tk\in T$).$M^{tk}[m]$包含至少一个协商原语 $NPrim^{tk}[m]$,大多情况包含一个 offer,即 $CC^{tk}[m]$.图 1(a)表示了协商线程的概念.在 *tk* 时刻,只有一个消息被发送,而由于协商中存在多个并行的协商线程,因此每个 *Thri* 中连续消息序列的时间 t_1,t_2,t_3 等并不一定在 *Time* 的顺序序列中是连续的.这里忽略发送和接收的时间差,因此,某消息的发送时间与接收时间认为是相同的.

(4) *DP* 是非静态有限阶段 Markov 决策过程集,每个参与协商的 Agent 拥有一个非静态有限阶段 Markov 决策过程,即 $DP=\{Dppa_1,Dppa_2,\ldots,DPpa_n\}$.整个协商过程由 *DP* 构成,当所有 $DPpa_i$ 都结束,协商过程也结束.$DPpa_i$ 为一个 5 元组(T,S,A,r,Q),其中,

1) *T* 是 Agent 决策时刻集.Agent 选取协商行动的时间点被称为决策时刻.决策时刻集是离散有限点集,$T=\{T_1,T_2,\ldots,T_n\}$.设定一个由离散线性顺序排列的时间点组成的全局时间 *Time*,则 $T\subset Time$.根据协商协议 *P*(如图 1 所示),Agent 有 5 个协商状态类别,其在每个协商状态类别都需做出相应的协商行动(见下面第 3)).因此,*T* 与协商状态类别相对应,即每当 Agent 到达某种协商状态时,都会对应一个决策时刻.注意,Ns_1 到 Ns_2 是一个有限循环,Agent 可能多次(有限次)到达这两个协商状态,因此,每种协商状态类别会对应多个决策时刻.

2) *S* 是 Agent 在所有决策时刻的环境状态集,$S=\{ST_1,ST_2,\ldots,ST_n\}$.Agent 的环境状态与决策时刻相关,在不同的决策时刻,其环境状态可能不同.环境状态涉及多种环境因素,如 Agent 可获取的资源、项目需求、市场状况等,因此有 $ST_i=\left(e_1^{T_i},e_2^{T_i},\ldots,e_k^{T_i}\right)(T_i\in T)$,其中 e_i 是影响建模过程和结果的环境因素.

本文将着重讨论 Agent 可获取的资源,考虑两种资源:协商剩余时间和参与协商的 Agent 数目,记为 e_1,e_2.将协商剩余时间分成两种,一种是时间充裕状况,协商剩余时间介于区间$[0,Deadline/2)$,记为 e_1^1,另一种是时间紧迫状况,协商剩余时间介于区间$[Deadline/2,Deadline]$(*Deadline* 是协商允许的最大时间),记为 e_1^1.类似地,参与协商的 Agent 数目也分为两类,一种为 Agent 数目多,介于区间$[\max,\max/2)$,记为 e_2^1,另一种为 Agent 数目少,介于区间$[\max/2,0]$,记为 e_2^2.max 是根据经验确定的针对当前任务类别的最大参与人数,也可以简化为开始参与协商的 Agent 数目,对于协商发起者,指的是协商响应者数目,对于协商响应者,指的是其他参与竞争的协商响应者数目.

3) *A* 是 Agent 在所有决策时刻与决策相关的可行动集,$A=\{AT_1,AT_2,\ldots,AT_n\}$.在每个决策时刻,存在策略 $\pi T_i:ST_i\to AT_i$.Agent 与决策相关的主要行动是产生和评价 offer,由此 π_{T_i} 包含确定将要发送的 offer 值的策略和对 offer 值进行评价的策略,分别记为 $\pi_{make\text{-}offer}^{T_i}$ 和 $\pi_{evalue\text{-}offer}^{T_i}$.$\pi_{make\text{-}offer}^{T_i}$ 通过基本战术、环境依赖战术与其权重确定,即 $\pi_{make\text{-}offer}^{T_i}=\left\{\rho,\left(tac1,w_{tac1}^{T_i}\right),\left(tac2,w_{tac2}^{T_i}\right),\ldots,\left(tacn,w_{tacn}^{T_i}\right)\right\}$ 确定,其中 ρ[18]是基本战术——在一定的可以接受的值范围内按照一定的让步率从最期望的值逐渐让步,*tacj* 是环境依赖战术,以环境因素为参数对基于基

本战术产生的 offer 进行调整,$w_{tacj}^{T_i}$ 为决策时刻 T_i 时 Agent 对 *tacj* 所确定的权重,$\sum_{j=1}^{n} w_{tacj}^{T_i}=1$.在每个决策时刻,Agent 可以通过对环境状态的观察动态确定各种战术的权重.本文重点讨论的两种环境因素所对应的战术分别为时间依赖战术τ和 Agent 资源依赖战术ς[18].令其权值分别为 w_τ和 w_ς,且 $w_\tau+w_\varsigma=1$,则我们按照 w_τ和 w_ς的取值将π*make-offer* 分为 3 种:(1) $w_\tau=w_\varsigma$,记为π*make-offer*1;(2) $w_\tau>w_\varsigma$,记为π*make-offer*2;(3) $w_\tau<w_\varsigma$,记为π*make-offer*3,体现了这两种环境因素对协商行为——offer 产生的不同影响,不同的π*make-offer* 对应不同的行为.Agent 需根据对环境状态的观察动态确定采用哪种π*make-offer*,即采取哪种行为.

$\pi_{evalue\text{-}offer}^{T_i}$ 由基于加法评分系统[19]的 offer 估价函数确定.令 offer 为 $cc=\{o_1,o_2,\ldots,o_n\}$,$V_{o_j}^{T_i}$ 为 Agent 在决策时刻 *Ti* 关于 o_j 的估价函数($V_{o_j}^{T_i}$ 单调增或单调减),当将所有 $V_{o_j}^{T_i}$ 的值域统一在某闭区间时,offer *cc* 的估价函数定义为 $V_{cc}^{T_i}=\sum_{o_j} w_{o_j}^{T_i} V_{o_j}^{T_i}$,其中 $w_{o_j}^{T_i}$ 为 Agent 在决策时刻 T_i 确定的 o_i 的权重,$\sum_{i=1}^{N} w_{o_i}=1$,则不同的 W_{o_i} 和 V_{o_i} 决定了不同的π*evalue-offer* 策略.与π*make-offer* 类似,π*evalue-offer* 也由 Agent 根据环境状态,通过调整 W_{o_i} 和 V_{o_i} 确定.这里,以协商主题的优先级变化导致其权重变化为例,将 W_{o_i} 的取值分成两类:(1) 所有 W_{oi} 值均相等,记为π *evalue-offer*1,即平等考虑所有协商主题;(2) 所有 W_{oi} 呈偏序排列,记为π*evalue-offe*2,即各种协商主题有某种优先级关系.

4) *r* 是 Agent 的行动报酬函数,$r: S\times A$ $(s\in S, a\in A)$.令 *R*(*s*,*a*)表示在状态 *s* 下采取行动 *a* 所得到的即时报酬,则若系统的环境状态有 *n* 种,行动有 *m* 种,*R*(*s*,*a*)将对应 *n*×*m* 个 *s*,*a* 对.根据上述有关环境状态的描述,本文着重讨论协商剩余时间 e_1,其类别有 2 种,与参与协商的 Agent 数目 e_2,其类别有 2 种,则环境状态有 2×2=4 种;又根据上述对 Agent 行动的描述,π*make-offer* 有 3 种, π*evalue-offer* 有 2 种,则 Agent 的行动有 3×2=6 种.由此相应的 *R*(*s*,*a*)有 24 个 *s*,*a* 对.每个 Agent 的具体报酬方案由与表 1 类似的形式给出(表 1 给出的是协商发起者的报酬函数,且限于篇幅仅给出了部分 *r*).

表 1 表明,对于协商发起者而言,当环境状态为(e_1^1,e_2^1)时,即协商时间充裕、参与协商的 Agent 数目充裕的情况下,采取行动(π*make-offer*1,π*evalue-offer*1),即生成 offer 时同等考虑协商时间和 Agent 数目、对 offer 进行评价时同等考虑所有协商主题,给与最高的报酬,而在(e_1^1,e_2^1)时采取其他行动,都会得到相对低的报酬;在(e_1^1,e_2^1)时,采取行动(π*make-offer*1,π*evalue-offer*2)得到的报酬将大于采取行动(π*make-offer*2,π*evalue-offer*1)得到的报酬,等等.

Table 1 Examples of partial reward function

表 1 报酬函数部分示例

S	*A*	*r*
(e_1^1,e_2^1)	(π*make-offer*1,π*evalue-offer*1)	*R*1
(e_1^1,e_2^1)	(π*make-offer*1,π*evalue-offer*2)	*R*2(<*R*1)
(e_1^1,e_2^1)	(π*make-offer*2,π*evalue-offer*1)	*R*3(<*R*2)
(e_1^1,e_2^1)	(π*make-offer*2,π*evalue-offer*2)	*R*4(<*R*3)
(e_1^1,e_2^2)	(π*make-offer*1,π*evalue-offer*1)	*R*2(<*R*1)
(e_1^1,e_2^2)	(π*make-offer*1,π*evalue-offer*2)	*R*5(<*R*2,*R*3)

基于报酬函数,可定义一个在任意决策时刻与策略相关的值函数 $V_i^{\pi}(s_i)=\sum_{t=i}^{N} E(R(s_t,\pi_t(s_t)))$ $(i\in T)$.其含义为从决策时刻 *i*、状态 s_i、采用策略π_i开始直至协商结束的总期望折扣报酬.那么最优策略是能使 $V_i^{\pi}(s_i)$ 最大的策略,记为*.根据经典 Bellman 最优等式[14],有

$$V_i^*(s_i)=\max_a\left\{R_i(s_i,a_i)+\sum_{t=i+1}^{N} p_i(s_t,a_i,s_i)V_t^*(s_t)\right\} \tag{1}$$

对所有 $s_i\in S, i\in T$,其中,$V_i^*(s_i)$被称为最优值函数,$p_i(s_t,a_i,s_i)$是系统在状态 s_i 执行 a_i 后转换到状态 s_t 的概率.Agent 通过 Q 学习算法找到*.

5) Q 是 Agent 选取最优策略的 Q 学习算法.Q 学习算法在不预知状态转换概率 $p_i(s_t,a_i,s_i)$的情况下,基于观察到的执行动作得到的报酬值和状态转化,迭代预测 $V_i^*(s_i)$值从而找到最优策略.

令 $Q_i(s_i,a_i)=R(s_i,a_i)+\sum_{t=i+1}^{N}p(st,a_i,s_i)V_t^*(s_t)$ 表示从状态 s_i、采取行动 a_i 开始,遵循策略最优策略*直至协商结束的总期望折扣报酬,代入公式(1),得到 $V_i^*(s_i)=\max_{a_i\in A_i}Q^*(s_i,a_i)$.因此,一旦已知 $Q(s_i,a_i)(s_i\in S,a_i\in A_i)$值,就能找到最优策略:

$$\pi^*(s_i)=\arg\max_{a_i\in A_i}Q^*(s_i,a_i)\ (i\in T) \tag{2}$$

在每一决策时刻 i,Agent 基于观察到的执行动作所得到的报酬 $R(s_t,a_t)$和状态变化 $i\to i+1$,通过下式迭代更新 $Q(s_i,a_i)$值:

$$Q(s_i,a_i)=Q(s_i,a_i)+\alpha(s_i,a_i)(R(s_t,a_t)+V^*(s_{i+1})-Q(s_i,a_i)) \tag{3}$$

从而维护 $Q(s_i,a_i)$值查找表(二维).其中$\alpha(s_i,a_i)$是随时间衰减的学习率,体现了预测值的变化率.

算法 1. AMNM-PA 的 Q 学习算法.

初始化 $Q(s_i,a_i)$值,对所有 $s_i\in S,a_i\in A_i$;其中对终止状态,即 $s_i=ST_n,Q(s_i,a_i)=0$;

给定 $R(s_i,a_i)$值,对所有 $s_i\in S,a_i\in A_i$;

1) 获取当前状态 s_i;

2) 根据公式(2)计算出$\pi^*(s_i)$,选择对应的 a_i;

3) 当 Agent 进入下一决策时刻 $i+1\in T$ 时,若 $i+1=Tn$,则转 5);

否则,获取当前状态 s_{i+1},根据公式(3)迭代更新 $Q(s_i,a_i)$值;

4) $\alpha=\alpha^*decay;i=i+1$;转 1);

5) 算法结束.

2 适应性多边协商算法

定义 1 给出了 Agent 在协商过程中何时(在哪些协商状态类别和相应的决策时刻)和如何采用 Q 学习算法选取策略从而选取行动以对变化的环境状态进行适应.据此可以分别针对协商发起者和协商响应者设计多边适应性协商算法,见算法 2 和算法 3.

算法 2. 协商发起者 a 的适应性多边协商算法.

初始化 *bestOffer*(最佳可接受 offer);初始化,$n=0,m=0$(交互轮次);

(1) 向其他 Agent $i(\in[1,N],N$ 为协商响应者数目)发送初始消息 $M_{a\to i}^{tk}[m](m=1)$ (tk 表示发送或接收时间,忽略同一消息发送和接收的时间差),包含原语 *Propose_CC* 和 offer $CC_{b\to i}^{tk}[m]$;令 *case*=1;

(2) 收到消息 $\{M_{i\to a}^{tk+1}[m]\}(i\in[1,N])$;

(3) 如果 $NPrim_{i\to a}^{tk+1}[m]$ ==*Terminate_N*,则终止协商线程 *Thri*;否则,按照 Q 学习算法找到 π_a^*,包含 π*make-offer*$_a^*$ 和 π*evalue-offer*$_a^*$;

如果 $NPrim_{i\to a}^{tk+1}[m]$ ==*Accept_CC*,根据 π*evalue-offer*$_a^*$ 比较 $CC_{i\to a}^{tk}[m]$ 与最佳 offer 的评价值,如果 $CC_{i\to a}^{tk+1}[m]$ 的价值大,则令 *bestOffer*= $CC_{i\to a}^{tk+1}[m]$,任务执行者=*i*,*flag*="*ending*";

如果 $NPrim_{i\to a}^{tk+1}[m]$ ==*Propose_CC*,1) 根据 π*make-offer*$_a^*$ 计算 $CC_{a\to i}^{tk+2}[m+1]$;2) 根据 π*evalue-offer*$_a^*$ 比较 $CC_{a\to i}^{tk+2}[m+1]$ 与 $CC_{i\to a}^{tk+1}[m]$ 的评价值;3) 如果 $CC_{a\to i}^{tk+2}[m+1]$ 的价值不比 $CC_{i\to a}^{tk+1}[m]$ 的价值大,则将 $CC_{i\to a}^{tk+1}[m]$ 列入可接受 offer 列表,令 *flag*="*ending*",且如果 $CC_{i\to a}^{tk+1}[m]$ 的价值比 *bestOffer* 的价值大,则令 *bestOffer*= $CC_{i\to a}^{tk+1}[m]$,任务执行者=*i*;4) 否则($CC_{a\to i}^{tk+2}[m+1]$ 的价值比 $CC_{i\to a}^{tk+1}[m]$ 的价值大),将 $CC_{a\to i}^{tk+2}[m+1]$ 列入发送列表;

(4) $n=n+1$;

(5) 如果 $n>=N$: 1) 令 $m=m+1$;2) 如果 *flag* ==“*ending*”,则向任务执行者发送原语 *Accept_CC*,向其他协商响应者发送 *Terminate_N*,进入(7);3) 如果 *flag*!=“*ending*”,则向协商响应者发送列表中相应的消息;4)令 $n=0$;清空发送列表;

(6) 如果 $n<N$,回到(2);

(7) 协商结束.

算法 2 表明,协商发起者每次收到所有协商响应者的回复后才能决定是否进行下一次交互.即此一对多协商相当于多轮次的拍卖.然而,传统拍卖,如公开拍卖,拍卖者给出的价格(即这里的 offer)是公开的,因此对于叫价者而言,得到的是相同的价格;而在我们的协商中,除了协商发起者首次发送的 offer 以外,其他每次交互向协商响应者发送的 offer 是不同的.

算法 3. 协商响应者 i 的适应性协商算法.

初始化 *bestOffer*(最佳可接受 offer);初始化 $m=0$(交互轮次);

(1) Agent i 收到 $M_{a\to i}^{tk}[m]$ ($i\in[1,N]$);

(2) 如果 $NPrim_{a\to i}^{tk}[m]$ == *Terminate_N*,则 Agent i 退出协商;如果 $NPrim_{a\to i}^{tk}[m]$ == *Accept_CC*,则进入(3);如果 $NPrim_{a\to i}^{tk}[m]$ ==*Propose_CC*,则:1) 按照 Q 学习算法找到 π_i^*,包含 π*make-offer*$_i^*$ 和 π*evalue-offer*$_i^*$;2) 根据 π*make-offer*$_i^*$ 计算 $CC_{i\to a}^{tk+1}[m]$; 3)根据 π*evalue-offer*$_i^*$ 比较 $CC_{i\to a}^{tk+1}[m]$ 和 $CC_{a\to i}^{tk}[m]$ 的评价值,$CC_{i\to a}^{tk+1}[m]$ 的价值不大于 $CC_{a\to i}^{tk}[m]$ 的价值,则向 Agent a 发送原语 *Accept_CC*;否则,向 Agent a 发送原语 *Propose_CC* 和 $CC_{i\to a}^{tk+1}[m]$;

(3) $m=m+1$;

(4) 协商线程结束.

3 应用及实验

本文提出的适应性协商方法实施于软件过程管理系统—— SoftPM[10,11]中.一个针对给定软件项目的软件过程模型是一个由 Agent 以及 Agent 之间达成一致的合同所组成的合同网,该模型在建模阶段用于 SoftPM 中项目经理的决策支持,在执行阶段用于项目参与人员的过程管理.在模型,即合同网的构建过程中,每一层的构建性能决定了模型整体的构建性能.因此,我们在 SoftPM 中针对实验室内部的一个小型项目,对形成层的一对多协商就引入了非静态 Markov 与 Q 学习算法的适应性协商与普通协商(即先前工作中未采用适应算法的协商)进行比较和分析,考察两种协商对该项目的过程模型生成的影响.

该项目需要两位来自学生的开发人员,组长负责项目的进度控制和一部分编码,组员负责另一部分的编码工作,两人进行交叉测试.组长已经确定.为此,组长 Agent *PM* 作为协商发起者与系统中的两个开发员 *M*1 和 *M*2 就编码任务的 3 个属性,即协商主题:价格(记为 *Prc*,取值范围[0,∞])、工期(记为 *Prd*,取值范围[0,∞])和质量等级(记为 *Qul*,取值范围[1,5],1 表示最高质量等级,5 则相反)进行协商,从而选出一个组员与之共同开发该项目.

下面给出两种协商的主要配置数据.

(1) 普通协商.表 2 给出了各 Agent 可接受的协商主题值范围,由此可得到各 Agent 对于 3 种协商主题的基本战术 ρ (定义见文献[18]).普通协商直接使用基本战术 ρ 产生 offer.根据表 2,取 3 个协商主题的评价函数值域均为[0,100],即无论代码量、工期和质量等级,当值对自身最有利时,对该值的评价分数为 100,最不利时为 0;为了简化,各 Agent 的协商主题评价函数均取直线函数(根据协商主题的不同性质可定义复杂的、不同的曲线函数);则可以得到各 Agent 对各协商主题的评价函数,如 *PM* 的 3 个估价函数分别为 $V_{prc}(prc)=-1\times prc+500$,$V_{prd}(prd)=-20\times prd+500$,以及 $V_{qul}(qul)=0,qul=1;25,qul=2;50,qul=3;80,qul=4;100,qul=5$.*M*1 和 *M*2 的协商主题的估价函数与 *PM* 类似但单调性相反.又根据表 3 给出的各 Agent 对不同协商主题的权值,采用加法评分系统,可以得到各 Agent 对 offer 的评价函数,如对于 PM,有 $V_{cc}=0.4\times V_{prc}+0.3\times V_{prd}+0.3\times V_{qul}$.

(2) 适应性协商.适应性协商在上述基本战术 ρ 的基础上,需要在每个决策时刻,根据对环境状态的观察,基

于 Q 学习算法找到适应该环境状态的$\pi_{make\text{-}offer}$,即从定义 1 中给出的 3 种$\pi_{make\text{-}offer}$中选取 1 种,以产生适应性的 offer.同样,适应性协商不采用上述非适应性协商的固定 offer 评价函数,而是在每个决策时刻,根据对环境状态的观察,基于 Q 学习算法找到适应该环境状态的$\pi_{evalue\text{-}offer}$,即从定义 1 中给出的两种不同的$\pi_{evalue\text{-}offer}$中选取一种,以适应性地评价 offer.本例中,Q 学习算法的学习率采用$\alpha(s_i,a_i)=\frac{\alpha_0}{N(s_i,a_i)}$,其中$\alpha_0$=1,$N(s_i,a_i)$为在状态 s_i 下行动 a_i 被选取的次数.具体协商过程见算法 2、算法 3.

Table 2 Ranges of acceptable negotiation objective values

表 2 可接受协商主题值范围

	PM	*M*1	*M*2
Prc (100RMB/day)	4-5	6-4	5-4
Prd (day)	20-25	30-20	25-15
Qul (level)	5-4	3-4	2-3

Table 3 Identified $\pi_{offer-evalue}$

表 3 指定的$\pi_{offer\text{-}evalue}$

	PM	*M*1	*M*2
Prc	0.4	0.5	0.6
Prd	0.3	0.2	0.2
Qul	0.3	0.3	0.2

实验主要考虑以下几项指标:

(1) 协商时间(negotiation time)和交互次数(interaction time):是指达成一致所需的协商时间和协商轮数(由于我们的实验没有在地理位置跨越较大或复杂的网络环境中进行,因此未考虑网络传输等方面的性能);

(2) 各 Agent 对达成一致合同的评价(contract evaluation):各 Agent 采用各自的评价函数对合同进行评价,体现了合同对单个 Agent 的价值(普通协商采用表 3 确定的评价函数,适应性协商在不同的决策时刻根据环境可能会采用不同的评价函数,因此这里的评价是基于协商结束阶段的评价函数);

(3) 对达成一致合同的联合评价值(coalition evaluation)[19].联合评价值是各 Agent 的合同评价值之和,体现了合同双方的共同利益,是软件过程模型——即由 Agent 组成的合同网的重要性能指标.

在实验中,Agent 采取适应性协商或普通协商中任一种协商方式,由此可以考察多种情况下,当各 Agent 采用不同协商方式时,整个协商过程的上述 4 种性能.这里给出其中的几种情况:

Case(1): *PM*,*M*1 和 *M*2 均采用非适应性协商;

Case(2): *PM* 采用适应性协商而 *M*1 和 *M*2 采用非适应性协商,中途 *M*1 退出;

Case(3): *PM* 和 *M*1 采用适应性协商而 *M*2 采用非适应性协商,中途 *M*1 退出;

Case(4): *PM* 和 *M*2 采用适应性协商而 *M*1 采用非适应性协商,中途 *M*1 退出;

Case(5): *PM*,*M*1 和 *M*2 均采用适应性协商.

实验结果见表 4.

Table 4 Partial experimental results

表 4 部分实验结果

	Negotiation time	Interaction time	Contract	Contract evaluation	Coalition evaluation
Case (1)	150	6	(500,25,4)	V_{PM}=24,V_{M1}=42	V_{PM}+V_{M1}=66
Case (2)	78	3	(423,23,3)	V_{PM}=57, V_{M2}=40	V_{PM}+V_{M2}=97
Case (3)	63	3	(423,23,3)	V_{PM}=57, V_{M2}=40	V_{PM}+V_{M2}=97
Case (4)	62	3	(408,22,3)	V_{PM}=70, V_{M2}=29	V_{PM}+V_{M2}=99
Case (5)	62	2	(408,22,3)	V_{PM}=70, V_{M2}=29	V_{PM}+V_{M2}=99

表 4 表明,*PM*,*M*1 和 *M*2 均采用适应性协商时(即 Case(5))的各种性能比它们均采用非适应性协商时(即 Case(1))的性能要高,如在协商时间上节省了 58%,在交互次数上减少了 66%,且尽管 *M*2 对 Case(5)达成一致的合同的评价 29 小于对 Case(1)达成一致的合同的评价 42,减少了 30%,但 *PM* 对 Case(5)达成一致的合同的评价 70 却大于对 Case(1)达成一致的合同的评价 24,增加了 191%,由此导致了 Case(5)的合同的联合评价 99 大于 Case(1)的联合评价 66,增加了 50%,即在合同对整体的意义上,Case(5)在很大程度上优于 Case(1).从 Case(1)与情况 Case(2)~Case(5)的性能对比上看,当参与协商 Agent 中越多地采用适应性协商时,性能越好,如 Case(2)一个 Agent 采用适应性协商在时间上的性能 78 比 Case(3)两个 Agent 采用适应性协商的性能 63 差.

该项目最终选择 Case(5)生成的过程,该过程包含:(1) *PM* 与 *M*2;(2) *PM* 与 *M*2 就编码任务达成的受合同值(408,22,3)约束的相互关系.项目按照该过程执行并顺利完成.由此可以看出,基于 AMNM-PA 的适应性协商能够

针对具体项目,以较好的性能生成全局利益优化的、在实际执行中合理的软件过程模型.

在本案例中,协商过程采用的是无人工参与的全自动协商过程.在应用中,也允许 Agent 所代表的软件开发人员参与协商过程,即当 Agent 给出 offer 后,其代表的软件开发人员可以对该 offer 进行调整后再发送给与其协商的其他 Agent.在这种情况下,环境因素变化对协商的影响更为显著.一个针对某项目的过程模型可能在全自动的情况下几分钟内被构建,也可能在人工参与的情况下花上一周或更长的时间.在这样一个时长内,考虑了多种环境因素的适应性协商更能给出优化的协商结果.

4 结论及未来工作

本文提出了一种用于软件过程建模的适应性多边协商模型 AMNM-PA,它以 Agent 表示软件过程中涉及的个体,包含组织、团队、个人等,通过 Agent 间的协商,动态、适应地建立针对给定软件项目的软件过程模型.AMNM-PA 基于 Markov 决策过程和 Q 学习,协商中的 Agent 通过对环境状态及状态-行动对的即时报酬的不断观察和学习(即 Q 学习),在每个决策点确定当前环境状态下能使期望报酬最高的协商策略和协商行为,从而建立变化的环境状态与协商过程及协商结果之间的适应关系,使得协商过程,即软件过程建模过程,以及协商结果,即所建立的软件过程模型具备一定的环境适应性.该适应性解决了软件过程建模受变化的资源、涉众需求、市场状况等环境因素的影响问题,且随着软件规模的不断加大,软件过程建模的复杂性、时间性也随之加大时,这种适应性的作用更为明显.AMNM-PA 已经在软件过程管理系统——SoftPM 中得到实施,支持针对具体软件项目的软件过程模型生成.应用结果表明,在环境变更较为频繁的项目中,AMNM-PA 能够为项目的任务分配者和任务承担者提供适应性的决策支持,最大限度地保证软件过程模型的合理生成.

本文下一步的工作将集中于 AMNM-PA 对软件过程模型执行中合同变更的支持.

References:

[1] Cugola G, Ghezzi C. Software processes: A retrospective and a path to the future. Software Process——Improvement and Practice, 1998,4(2):101−123.

[2] Phongpaibul M, Koolmanojwong S, Lam A, Boehm B. Comparative experiences with electronic process guide generator tools. In: Wang Q, *et al*., eds. Proc. of the ICSP 2007 Software Process Dynamics and Agility. LNCS 4470, Berlin, Heidelberg: Springer-Verlag, 2007. 61−72.

[3] Yu ESK, John Mylopoulos. Understanding "Why" in software process modelling, analysis, and design. In: Proc. of the 16th Int'l Conf. on Software Engineering. 1994. 159−168. http://ieeexplore.ieee.org/xpl/freeabs_all.jsp?tp=&arnumber=296775&isnumber=7343

[4] Ambriola V, Ciancarini P, Montangero C. Software process enactment in Oikos. ACM SIGSOFT Software Engineering Notes, 1990,15(6):183−192.

[5] Yilmaz L, Phillips J. Organization-Theoretic perspective for simulation modelling of agile software processes. In: Wang Q, *et al*., eds. Proc. of the SPW/ProSim 2006. LNCS 3966, Berlin, Heidelberg: Springer-Verlag, 2006. 234−241.

[6] Turetken O, Demirors O. An approach for decentralized process modeling. In: Wang Q, *et al*., eds. Proc. of the ICSP 2007 Software Process Dynamics and Agility. LNCS 4470, Berlin, Heidelberg: Springer-Verlag, 2007. 195−207.

[7] Boehm B. The future of software processes. In: Li M, *et al*., eds. Proc. of the SPW 2005 Unifying the Software Process Spectrum. LNCS 3840, Berlin, Heidelberg: Springer-Verlag, 2005. 10−24.

[8] Zhao XP, Li MS, Wang Q, Chan K, Leung H. An agent-based self-adaptive software process model. Journal of Software, 2004,15(3):348−359 (in Chinese with English abstract). http://www.jos.org.cn/1000-9825/15/348.htm

[9] Zhao X, Chan K, Li M. Applying agent technology to software process modeling and process-centered software engineering environment. In: Proc. of the 2005 ACM Symp. on Applied Computing (SAC 2005). New York: ACM Press, 2005. 1529−1533. http://portal.acm.org/citation.cfm?id=1066677.1067021

[10] Wang Q, Xiao J, Li M, Nisar MW, Yuan R, Zhang L. A process-agent construction method for software process modeling in SoftPM. In: Wang Q, *et al*., eds. Proc. of the SPW/ProSim 2006 Software Process Change. LNCS 3966, Berlin, Heidelberg: Springer-Verlag, 2006. 204−213.

[11] Zhang L, Wang Q, Xiao J, Li J, Xie L, Li M. A tool to create process-agents for oec-spm from historical project data. In: Wang Q, *et al*., eds. Proc. of the ICSP 2007 Software Process Dynamics and Agility. LNCS 4470, Berlin, Heidelberg: Springer-Verlag, 2007. 84−95.

[12] Zeng D, Sycara K. Bayesian learning in negotiation. Int'l Journal of Human-Computer Studies, 1998,48(1):125−141.

[13] Littman ML. Markov games as a framework for multi-agent reinforcement learning. In: Proc. of the 11th Int'l Conf. on Machine Learning (ML-94). Morgan Kaufmann Publishers, 1994. 157−163. http://citeseerx.ist.psu.edu/viewdoc/ summary?doi=10.1.1.48.8623

[14] Hu J, Wellman MP. MultiAgent reinforcement learning: Theoretical framework and an algorithm. In: Proc. of the 15th Int'l Conf. on Machine Learning (ML-98). Morgan Kaufmann Publishers, 1998. 242−250. http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.42.1313

[15] Garcia F, Ndiaye SM. A learning rate analysis of reinforcement learning algorithms in finite horizon. In: Proc. of the 15th Int'l Conf. on Machine Learning (ML-98). Morgan Kaufmann Publishers, 1998. 215−223. http://citeseerx.ist.psu.edu/viewdoc/summary?doi= 10.1.1.48.3867

[16] Narayanan V, Jennings NR. An adaptive bilateral negotiation model for E-Commerce settings. In: Proc. of 7th Int'l IEEE Conf. on E-Commerce Technology. 2005. 34−39. http://ieeexplore.ieee.org/Xplore/login.jsp?url=/iel5/10218/32584/01524026.pdf?temp=x

[17] Li J, Yahyapour R. Learning-Based negotiation strategies for grid scheduling. In: Proc. of 16th IEEE Int'l Symp. on Cluster Computing and the Grid (CCGRID 2006). 2006. 576−583. http://ieeexplore.ieee.org/Xplore/login.jsp?url=/iel5/10856/34197/01630873.pdf?tp=&isnumber=&arnumber=1630873

[18] Li N, Wang Q, Li M, Du S, Xiao J. A multilateral negotiation method for software process modelling. In: Wang Q, *et al*., eds. Proc. of the ICSP 2007 Software Process Dynamics and Agility. LNCS 4470, Berlin, Heidelberg: Springer-Verlag, 2007. 147−158.

[19] Raiffa H, Wrote; Song X, Sun XX, Trans. The Art and Science of Negotiation. Bejing: Beihang University Press, 1987. 152−170 (in Chinese).

附中文参考文献:

[8] 赵欣培,李明树,王青,陈振冲,梁金能.一种基于 Agent 的自适应软件过程模型.软件学报,2004,15(3):348−359. http://www.jos.org.cn/1000-9825/15/348.htm

[19] Raiffa H,著;宋欣,孙小霞,译.谈判的艺术与科学.北京:北京航空航天大学出版社,1987.152−170.

黎嵬(1975－),女,海南三亚人,博士,主要研究领域为 Agent 协商,软件过程技术,业务过程建模,Web 服务.

赵琛(1967－),男,博士,研究员,博士生导师,CCF 高级会员,主要研究领域为编译技术及应用,软件测试方法和工具.

李明树(1966－),男,博士,研究员,博士生导师,CCF 高级会员,主要研究领域为软件工程方法学,软件过程技术,需求工程,软件工程经济学.

杜栓柱(1971－),男,博士,副研究员,主要研究领域为业务过程建模,Web 服务,知识管理.

王青(1964－),女,博士,研究员,博士生导师,CCF 高级会员,主要研究领域为软件过程技术.