面向预定义过程的强化学习WS组合

付燕宁,张家臣,刘 磊

(吉林大学 计算机科学与技术学院,长春 130012)

摘 要:将强化学习应用到基于预定义过程的WS组合,提出了QoS实际性能驱动的随机强化学习Web服务组合方法。利用该方法可以根据WS的实际性能逐渐为模型中的任务学习到优化的任务分配策略,根据任务分配策略所选择的WS的协同执行能最大程度地满足用户的服务请求。与同类方法相比,该方法根据服务的QoS实际性能将WS以往性能数据的利用与持续不断地对新的优化组合的探索融合到一起,逐渐学习到与过程模型相对应的优化服务组合。通过对熵取值范围的讨论,说明了对以往策略的利用与持续不断探索之间的关系。

关键词:计算机应用;组合服务;预定义过程;强化学习;服务选择;合成器

中图分类号:TP393.09 **文献标志码**:A **文章编号**:1671-5497(2010)05-1313-05

Process-oriented WS composition via reinforcement learning

FU Yan-ning, ZHANG Jia-chen, LIU Lei

(*College of Computer Science and Technology, Jilin University, Changchun* 130012, *China*)

Abstract: A randomized reinforcement learning approach for Web Service (WS) composition is proposed to determine the optimal WS to execute the task in a predefined process model. This approach can learn the optimal task allocation policy corresponding to the task in a given model based on actual performance of WS. It can also identify the optimal WS corresponding to the task in a given model based on task allocation policy. By this approach, the coordinated execution can continually satisfies user's service request to the optimal levels. Compared with other similar methods, the approach integrates the exploitation of acquired data about the past performance of individual services with the optimal, undirected, continual exploration of new composition according to QoS actual performance so as to learn the optimal WS composition corresponding to the process method. The relation between exploitation and exploration is accounted for discussing the range of the entropy.

Key words: computer application; composite service; predefined process; reinforcement learning; service choice; composer

随着WS的大量出现,WS组合的主要问题不再是能否找到所需要的WS,而是如何从满足功能需求的WS集合中选择所需要的WS问题[1]。很多研究工作没有很好地解决组合服务与动态变化Web环境相适应的问题,导致组合服务性能的退化或失效。文献[2]着重解决满足马尔可夫性质的不完全信息下的服务选择问题,但该方法不能保证在动态变化环境中WS组合的有效

收稿日期:2009-06-05.

基金项目:国家自然科学基金项目(60873044).

作者简介:付燕宁(1965-),男,副教授,博士.研究方向:面向服务的计算,本体及本体工程.

E-mail:fuyn@jlu.edu.cn

性。文献[3]虽然考虑了动态改变的 Web 环境给服务组合带来的影响,但是将每次组合服务的执行都当成一个新的 WS 组合问题,这样每次组合时都要探索新的组合方案,没有考虑到利用过去形成的成熟的解决方案。文献[4]针对 WS 的 QoS 估计值与真实值之间的较大偏差降低服务组合成功率问题,提出了一种自适应 QoS 管理体系结构,尽管采用马尔可夫决策过程(Markov desition process,MDP)解决了动态 Web 环境影响组合服务可靠性的问题,但是所采用的方法是一次探索而不是在线持续探索,没有考虑对组合服务再利用问题。文献[5]提出通过为每个 WS 指定的信任值选择服务,这种方法涉及到了对以往 WS 性能历史记录的利用,但是没有考虑对动态 Web 环境的探索。文献[6]提出了一种用于自动计算的分散多 agent 系统架构,这种架构提供了注册表用以保存 agent 可执行的一组任务。他们假定已发布的性能就是实际的性能,没有利用有关 agent 实际性能的先验数据,因此削弱了对开放的、动态变化环境的适应性。尽管有些研究[4,7]采用学习机制来解决在开放的 Web 环境中的 WS 选择问题,但是没有考虑对动态变化的 Web 环境持续探索的问题。

文献[8]提出了一个探索和利用相集成的模型,本文将这种模型进行了调整,应用于预定义过程的 WS 选择问题,用于解决动态变化的 Web 环境中对串型服务请求模型中的服务选择问题。本文提出的强化学习组合算法通过对过去观察到的 WS 性能的利用,以及对 WS 及其性能的变化的探索,逐步逼近一套最优化的 WS 选择策略。该方法与以往方法的不同在于:在每次 WS 组合过程中都能根据环境的变化不断地调整服务选择策略,同时还能最大程度地利用以往形成的组合策略。

1 问题描述

每当合成器尝试选择了一个 WS,该 WS 使环境转换到一个新的任务状态并且引发一个反馈,称作"奖赏",该奖赏反映了所尝试的 WS 的性能。合成器可以根据服务的各种质量参数、期限、信誉、成本以及用户偏好等作为计算服务奖赏的依据,合成器据此判定 WS 的优劣来选择优化的服务。Maximilien 和 Singh 在文献[5]中使用"信任"从一组执行同样任务的服务中选择优化的服务,并且利用使用者给出的评价计算信誉,本文选择某种 QoS 性能参数的信誉度作为选择优化服务的依据,通过比较由合成器所观察到的 WS 的实际性能和 WS 提供者发布的性能来计算信誉。

定义 1 对于服务 w,其 QoS 参数 k 的信誉为

$$r_k(w)=\frac{1}{n}\sum_{i=1}^{n}\left[(\hat{v}_k^{adv}-v_k^i)^2\,\delta^{-time(v_k^i)}\right]$$

式中:$\hat{v}_k^{adv}$ 为参数 k 的发布值;v_k^i 为参数 k 的第 i 次实际观察值;$time(v_k^i)$ 为返回观察的时间(对于最新观察值,该函数值为 1,对于其他观察值,该函数值大于 1);δ 为给定质量参数的衰减因子,控制观察值对信誉度的影响,观察值越老,对信誉的影响越小。

定义 2 代价函数 $c(t,w)$ 表示将服务 w 分配给任务 t 所付出的代价,$c(t,w)$ 实质上是函数 $r_k(w)$,即

$$c(t,w)=r_k(w)$$

代价函数的最小值与性能参数 k 的优化值相对应,也就是说,函数值越小,服务的发布值就越接近其观察值,服务就越优化。

定义 3 设服务请求模型包括 m 个串行任务,对于每一个任务 $t(1\leqslant t\leqslant m)$ 都存在一个任务分配状态 $k(1\leqslant k\leqslant m)$,任务分配状态图(Task allocation graph,TAG)如图 1 所示,表示为 TAG = (K,E,W),其中:①$K=\{1,2,\cdots,m,m+1\}$,$m+1=d$ 表示无代价的任务分配结束状态。②$E=E_1\cup E_2\cup\cdots\cup E_m$,设可以分配给任务 t 的服务个数为 n,则 $E_i=\{(t,w_{ij})\mid 1\leqslant j\leqslant n\}$。两个状态之间有多少条边,就表明针对同一个任务 t 有多少种可能的分配(本文将满足同一个任务需求的所有服务称候选服务集,记作 S(k))。③$c(t,w_{ij})\in W$,表示边(t,w_{ij})上的权重,也就是将任务 t 分配给服务 w_{ij} 的代价。

优化 WS 组合问题实际上就是合成器为过程模型中的任务选择优化 WS 的问题。也就是说,针对任务分配状态 $k(k=1,2,\cdots,m)$,合成器根据由代价所计算出来的某种策略,为过程模型中

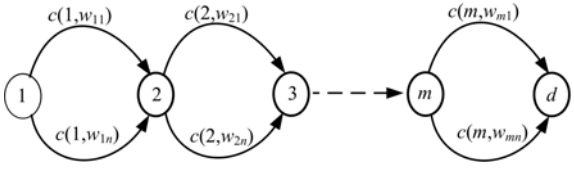

图 1 任务分配状态图 TAG

Fig. 1 Task allocation graph

的任务 $t(t=1,2,\cdots,m)$ 从与之相对应的候选服务集 $S(k)$ 中选择 WS 的问题。为过程模型中的任务选择 WS 实质上是随机顺序决策问题，MPD 是解决顺序决策问题常用的数学模型。当 MPD 模型中的状态转移概率矩阵未知时，可以通过学习技术采用试错的方法学习最优策略。强化学习(Reinforcement learning)通过值迭代或策略迭代的方式逼近 MPD 中的最优策略，其通过试错的方法学习从状态 k 到 $S(k)$ 之间的映射，使从初始状态 1 到终止状态 d 的累计代价达到最小。根据 Bellman 最优定理，对于状态空间有限的单链 MPD 问题，必然存在一个最优策略[9]；同样，对于任务分配问题也必然存在一个最优的任务分配策略 $\Pi\equiv\{\pi_k(w),k=1,2,\cdots,m\}$，这个最优策略是根据任务分配的代价逐步学习到的。当合成器根据过程模型进行 WS 组合时，根据任务分配策略 Π 将每个任务 t 映射到 $S(k)$ 上。这样，为过程模型中的任务选择 WS 问题实质上就转化为寻找任务分配策略 Π 的问题。

2　优化策略的计算

强化学习的一个主要特点是：它以集成方式明确地处理探索/利用问题，以及概率分布的在线估计问题。这样，探索/利用之间的权衡问题可以说是全局优化问题：即一方面找到使预期累计代价最小化的探索策略，另一方面又能保证在同一状态上的固定探索程度。换句话说，在持续探索的情况下，还要使利用达到最大化。将强化学习应用于任务分配要解决 3 个问题：①如何量化探索；②如何控制随机选择；③如何在给定的探索度下计算优化的策略。

2.1　确定探索度

为了控制探索，对每个状态都要定义熵。探索度是由香农熵来度量的，香农熵与将任务分配给 WS 的概率分布相关。文献[8]给出了在某一状态的任务分配概率分布的熵，本文用它来表示在状态 k 为任务选择 WS 概率分布的熵。

定义 4　状态 k 的探索度 E_k 被量化为

$$E_k=-\sum_{w\ni S(k)}\pi_k(w)\log\ \pi_k(w)$$

式中：E_k 表示在状态 k 选择 WS 的不确定性，用来控制合成器在 k 的探索。

当不存在不确定性时，熵等于 0；当不确定性程度最大时，$\pi_k(w)=1/n_k$（是一种平均分布），熵等于 $\log(n_k)$，其中 n_k 表示在状态 k 可供分配的 WS 数量。

定义 5　探索率 $ER_k\in[0,1]$ 是 E_k 实际值与其最大值之间的比率，$ER_k=E_k/\log(n_k)$。

确定某个状态的熵就确定了该状态的探索级别。增大熵值就是增加了探索，当熵达到最大值时，探索就不存在了，完全可以在随机地(使用平均分布)、不考虑代价的情况下选择 WS。可以利用探索率来控制合成器的探索级别。因此，随机强化学习 WS 组合方法就是在保证对每一个状态 k 一定的探索级别的情况下，使从初始状态 1 到终止状态 d 的所有路径上的累计代价 $U_\pi(1)$ 达到最小。

$$U_\pi(1)=E_\pi\Big[\sum_{k=1}^{d}c(k,w)\Big]$$

期望值 E_π 根据策略 Π 取值，即根据在状态 k 的服务的随机选择取值。

2.2　优化策略的计算

令代价和策略以及累计代价初始值分别为 $\hat{c}(k,w)=0$、$\hat{\pi}_k=1/n_k$、$\hat{U}(k)=0$。在状态 $k(k=1,2,\cdots,m)$，合成器利用概率分布将任务 t 分配给服务 w，并获得相应的代价 $c(k,w)$(代价函数在动态环境下随着时间的变化而变化)；合成器更新 $\hat{c}(k,w)$、$\hat{\pi}_k(w)$ 和 $\hat{U}(k)$，随后合成器进入到新状态 $k+1$，重复这个过程直至状态 d。

(1)初始阶段

令 $\hat{U}(d)=0$，由于 d 是目标状态，因此目标状态的代价为 0。

(2)在一定探索条件下，计算任务分配策略和累计代价。

对于任意一个状态 $k(k=1,2,\cdots,m)$，计算：

①利用当前概率估计值将任务 t 分配给服务 w，并利用当前代价 $c(k,w)$ 更新其估计值 $\hat{c}(k,w)$。

$$\hat{c}(k,w)\leftarrow c(k,w)$$

②更改状态 k 的累计代价。

$$\hat{U}(k_i)=\sum\hat{\pi}_k(w)[\hat{c}(k,w_i)+\hat{U}(k'_i)],\quad w_i\in S(k)$$

$$\hat{U}(d)\leftarrow 0,d\text{ 是目标状态}$$

式中：$\hat{U}(k_i)$ 为在状态 k 选择了服务 w_i 到目标状态的累计代价；$k'_i=f_k(w_i)$ 为在状态 k 选择服务

w_i 所到达的状态,并且 $k \neq d$。

③将状态 k 的概率分布更改为

$$\hat{\pi}_k(w_i) = \frac{\exp\{-\theta_k[\hat{c}(k,w_i)+\hat{U}(k'_i)]\}}{\sum_{w_j \in S(k)} \exp\{-\theta_k[\hat{c}(k,w_j)+\hat{U}(k'_j)]\}}$$

式中:θ_k 要根据事先定义的每个状态的熵 E_k 来确定。由于函数 $\theta_k(E_k)$ 是严格单调递减的,因此用线性搜索算法(例如折半查找)可以找到与给定熵值 E_k 相对应的 θ_k[10]。这就说明在保持一定的探索率下,任务分配的概率分布可以使从开始状态到终止状态的累计代价最小。

2.3 优化策略的计算算法

合成器尝试各种不同的任务分配,并从每一种分配的结果进行学习,逐步学习到最优化的任务分配策略。设 G 为针对一个给定的过程模型的任务分配图以及所有可能的分配;*entropy* 为服务请求者给出的探索率;k 为任务分配问题的当前状态,是任务分配图中的一个结点;$\hat{\pi}$ 为合成器所学习到的当前状态的策略;$\hat{U}(k)$ 为合成器在状态 k 所计算出来的当前状态的累计代价;$\hat{c}(k,w)$ 为在状态 k 选择了服务 w 的执行代价。合成器学习任务分配决策的值迭代以及学习任务分配决策的算法如下:

Agorithm: optimalPolicy(G,entropy)

1 begin

Initialization

k←initialState(G)

$\hat{\pi}$←getPolicy(G)

$\hat{U}$←getValueiteration(G)

$\hat{c}$←getCostMatrix(G)

2 if(k∉terminalState(G))

w←chooseServices($\hat{\pi}$,k)

cost←observeCurrentCost(w)

3 $\hat{c}$(k,w)←cost

4 $\hat{U}(k) \leftarrow \sum_{w \in U(k)} \hat{\pi}_k[\hat{c}(k,w)+\hat{U}(k')], k'=f_k(w)$

θ_k←computeTeta(entropy,$\hat{\pi}$)

for each services $w_i \in S(k)$

5 $\hat{\pi}_k(w_i) = \frac{\exp[-\theta_k(\hat{c}(k,w_i)+\hat{U}(k'_i))]}{\sum_{w_j \in S(k)} \exp[-\theta_k(\hat{c}(k,w_j)+\hat{U}(k'_j))]}$

end for

6 k←observeCurrentState(G)

7 Goto 2

8 endif

9 end

合成器在不同的时刻与不同的任务分配状态(某一时刻所处的环境)交互,也就是说合成器从状态 1 开始进行任务分配直至终止状态 d。每当合成器在状态 k 根据策略 $\hat{\pi}$ 选择了一个执行任务的 WS 时,就产生一个相应的代价(在动态环境下随着时间的改变而改变)。合成器利用该代价更新相应的累积代价和策略,据此学习到了这种分配的优劣,随后合成器进入一个新的任务分配状态 $k+1$。通过一系列交互过程,合成器逐步学习到了一套任务分配策略。

3 算法分析

由算法中步骤 5 可以看出,当 θ_k 非常大时,$\pi_k(w) \approx 0$,熵 E_k 几乎为 0;在这种情况下,利用 $(c(k,w_i)+\hat{U}(k'_i))$ 的最小值作为选择 WS 的概率。换句话说,如果服务群 $S(k)$ 中有服务 w_i,$\pi_k(w_i) \approx 1$,则其他服务 $w_j(j \neq i)$ 的 $\pi_k(w_j) \approx 0$。因此,步骤 4 就可以改写为

$$\hat{U}(k) \leftarrow \min[c(k,w_i)+\hat{U}(k'_i)$$
$$w_i \in S(k), k'_i = f_k(w_i), k \neq d$$
$$\hat{U}(d) \leftarrow 0$$

式中:d 为目标状态。

当所有状态的探索率为 0 时,非线性方程就退化为 Bellman 方程,利用 Bellman 方程查找从初始状态到目标状态的最短路径。随着 Web 环境的变化,尽管有的服务不存在了,或者出现了代价更低的服务,合成器也始终利用这个策略选择服务。这种策略只有对以往策略的利用,而不再探索新的服务组合,不能适应 WS 及其性能不断变化的特点。

当 $\theta_k = 0$ 时,概率分布退化为 $\pi_k(w) = 1/n_k$,熵值 $E_k = \log(n_k)$,所有状态的探索度达到最大值。在这种情况下,非线性方程退化为线性方程,此时在马尔可夫链中利用均为 $1/n_k$ 的转移概率计算从初始状态到目标状态的累计代价。换句话说,合成器此时不考虑代价因素,随机地选取任何一个服务进行任务分配。这种策略总是处于随机的探索状态,而没有对以往所学习到的策略的利用。

当 $0 < E_k < \log(n_k)$,合成器可以学习到优化

的探索与利用的策略，既能保持一定的探索，又能使利用最大化。由于函数 $\theta_k(E_k)$ 是严格单调递减的，因此，随着熵 E_k 的增大(随着 θ_k 减小)，探索增加，利用减少。

通过上述分析可知，合理地给定熵值 E_k，该算法就可以逐步学习到一个优化的任务分配策略，既能保持对每一个状态的一定程度的探索，又能充分利用由以往性能参数所学习到的分配策略。每当 Web 环境发生变化时，该算法都能动态地调整任务分配策略，即使 Web 环境未发生变化，该算法也能根据 WS 的实际性能的变化调整任务分配策略。

4　结束语

针对在动态的 Web 环境下如何为过程中的任务选择服务的问题，将基于过程 WS 组合转化为强化学习，提出了基于强化学习的 WS 组合方法。该方法可以实现对动态变化的 Web 环境的持续探索，同时还能最大程度实现对过去形成组合策略的利用，从而使所形成的组合服务具有较强的适应性，提高组合服务的可靠性。本文对马尔可夫决策过程(MDP)[11]模型推广使用，使之表示 WS 组合过程中的任务分配问题；将文献[8]给出的强化学习算法进行了调整和扩充，将其应用于为预定义过程中的任务选择执行任务的 WS；并且这种服务选择策略是建立在所观察到的 WS 的实际性能上，而不是建立在服务提供者所发布的性能上。

本文提出的算法还存在以下有待解决的问题：需要人工预先给定各个状态的探索率来确定合成器对状态的探索，确定适当的探索率是值得考虑的问题；需要通过在线连续探索逼近最优化的服务组合，如何界定最优化的服务组合有待探讨；该组合算法尚未经过实际验证，支持用户最终编程的服务组合也是一项重要的工作。支持用户最终编程的服务组合涉及到许多基础性研究工作，如服务 QoS 属性和功能属性相集成的服务描述语言，包含任务功能属性和 QoS 属性的服务组合模板等。

参考文献：

[1] 史玉良，张亮，施伯乐. 一种选择最优 Web 服务的方法[J]. 小型微型计算机系统，2007，28(4)：720-724.
Shi Yu-liang，Zhang Liang，Shi Bo-le. Method to select the optimal web services[J]. Journal of Chinese Computer Systems，2007，28(4)：720-724.

[2] 陈彦萍，李增智，唐亚哲，等. 一种满足马尔可夫性质的不完全信息下的 Web 服务组合方法[J]. 计算机学报，2006，29(7)：1076-1083.
Chen Yan-ping，Li Zeng-zhi，Tang Ya-zhe，et al. A method satisfying markov process of web service composition under incomplete constrains[J]. Chinese Journal of Computers，2006，29(7)：1076-1083.

[3] Zeng L Z，Benatallah B，Dumas M. Quality driven Web service composition[C]// Proc of the WWW 2003. Budapest：ACM，2003.

[4] 范小芹，蒋昌俊，王俊丽，等. 随机 QoS 感知的可靠 Web 服务组合[J]. 软件学报，2009，20(3)：546-556.
Fan Xiao-qin，Jiang Chang-jun，Wang Jun-li，et al. Random-QoS-aware reliable web service composition[J]. Journal of Software，2009，20(3)：546-556.

[5] Maximilien E M，Singh M P. Multi-agent system for dynamic web services selection[C]// Proc Int Conf Auton Agents and Multi-Agent Syst，Utrecht，2005.

[6] Tesauro G，Chess D M，Walsh W E，et al. A multi-agent systems approach to autonomic computing[C]// Proc Int Conf Auton Agents Multi-Agent Syst. Washington，DC，USA：IEEE Computer Society，2004.

[7] Abdallah S，Lesser V. Modeling task allocation using a decision theoretic model[C]// Proc Int Conf Auton. Agents and Multi-Agent Syst. New York，USA：ACM，2005.

[8] Achbany Y，Fouss F，Yen L，et al. Optimal tuning of continual online exploration in reinforcement learning[C]// Proceedings of the International Conference on Artificial Neural Networks. German：Springer Berlin/Heidelberg，2006.

[9] 高阳，周如益，王皓，等. 平均奖赏强化学习算法研究[J]. 计算机学报，2007，30(8)：1372-1378.
Gao Yang，Zhou Ru-yi，Wang Hao，et al. Study on an average reward reinforcement learning algorithm[J]. Chinese Journal of Computers，2007，30(8)：1372-1378.

[10] Bazaraa M S，Sherali H D，Shetty C M. Nonlinear Programming：Theory and Algorithms[M]. New York，USA：John Wiley and Sons，1993.

[11] Sutton R S，Precup D，Singh S P. Between MDPs and semi-MDPs：a framework for temporal abstraction in reinforcement learning[J]. Artificial Intelligence，1999，112(1/2)：181-211.