计算机系统与计算机网络中的动态优化：模型、求解与应用

林 闯[1] 万剑雄[2] 向旭东[2] 孟 坤[2] 王元卓[3]

[1](清华大学计算机科学与技术系 北京 100084)

[2](北京科技大学计算机与通信工程学院 北京 100083)

[3](中国科学院计算技术研究所 北京 100190)

摘 要 动态优化是计算机系统与计算机网络中进行资源分配与任务调度等方面研究所采用的主要理论工具之一.目前,国内外已开展大量研究,致力于深化动态优化的理论研究与工程应用.文中从模型、求解与应用3个角度,对马尔可夫决策过程动态优化理论模型进行了综述,并重点介绍了将动态优化理论与随机Petri网理论相结合的马尔可夫决策Petri网和随机博弈网模型,详细讨论了这些模型的建模方法、求解算法与一些应用实例.最后,对全文进行了总结,并对未来可能的研究方向进行了展望.

关键词 动态优化;马尔可夫决策过程;随机Petri网;马尔可夫决策Petri网;随机博弈网

中图法分类号 TP393 **DOI号**: 10.3724/SP.J.1016.2012.01339

Dynamic Optimization in Computer Systems and Computer Networks: Models, Solutions, and Applications

LIN Chuang[1] WAN Jian-Xiong[2] XIANG Xu-Dong[2] MENG Kun[2] WANG Yuan-Zhuo[3]

[1](*Department of Computer Science and Technology, Tsinghua University, Beijing* 100084)

[2](*School of Computer & Communication Engineering, University of Science & Technology Beijing, Beijing* 100083)

[3](*Institute of Computing Technology, Chinese Academy of Sciences, Beijing* 100190)

Abstract Dynamic optimization is one of the most popular theoretical tools to study resource allocation and task scheduling problems in computer systems and computer networks. At present, a vast number of researches have been on their way to enhance the theoretical basis and extend the industrial applications of dynamic optimization theory. This paper provides an overview of Markov Decision Process (MDP) from the perspectives of models, solutions, and applications. We also survey two types of extended dynamic optimization models, i.e., Markov Decision Petri Nets (MDPN) and Stochastic Game Nets (SGN), which combine dynamic optimization theory and stochastic Petri nets theory. We focus on the model construction, solution techniques, and applications of these models. Finally, we discuss some possible research challenges in the future.

Keywords dynamic optimization; Markov decision processes; stochastic Petri nets; Markov decision Petri nets; Stochastic Game nets

收稿日期:2012-05-03;最终修改稿收到日期:2012-06-15.本课题得到国家"九七三"重点基础研究发展规划项目基金(2010CB328105,2009CB320505)、国家自然科学基金重点项目(60932003)和国家自然科学基金面上项目(61070182,60973144,60973107,61173008,61070021)资助.**林 闯**,男,1948年生,博士,教授,博士生导师,主要研究领域为计算机网络、系统性能评价、安全分析和随机Petri网.E-mail: chlin@tsinghua.edu.cn. **万剑雄**,男,1982年生,博士研究生,主要研究方向为性能评价和最优控制. **向旭东**,男,1986年生,博士研究生,主要研究方向为性能评价和最优控制. **孟 坤**,男,1980年生,博士研究生,主要研究方向为性能评价和随机模型. **王元卓**,男,1978年生,博士,副教授,主要研究方向为随机Petri网与网络安全.

1 引 言

随着计算机网络与计算机系统在国民生活各个领域应用的不断拓展,其承载的业务种类与数量也在不断增加.如何在复杂的应用环境中合理地分配系统资源并进行调度任务,以提高计算机系统与计算机网络的运行效率,降低运行成本,是一个亟待解决的问题.

优化理论是学术界研究计算机系统与计算机网络中资源分配与任务调度问题普遍采用的方法之一.从时间这个维度进行分类,优化理论可分为静态优化与动态优化两种.其中,静态优化将系统看作为一个时不变系统,即将系统的资源需求量与资源保有量视为一个与时间无关的常量.但是,实际的系统往往都是随时间变化的,而且会受到各种外部随机事件的影响.静态优化模型忽略了未来可能的系统变化,也不能反映决策者当前行为对未来的影响,无法刻画系统随时间变化的特性.因此,本文着重研究动态优化理论在计算机系统与计算机网络中的应用.在动态优化理论中,系统的目标函数是系统收益关于时间的累积量.相对于静态优化理论,动态优化理论可以较好地对系统的时变性进行刻画,更好地反映系统当前决策对时间累积目标函数的影响.

动态优化的基本理论模型是马尔可夫决策过程(Markov Decision Process,MDP).MDP 可以用来描述这样一类离散时间决策过程:系统 $t+1$ 刻状态的转移,只依赖于 t 时刻的系统状态与决策者的行为,而与$[0,t-1]$时间段内的系统状态与决策者行为无关.MDP 可以从执行时间、决策者观测能力、状态转移的确定关系、时间的连续性、状态转移/收益的确定性、是否具有附加限制条件以及决策目标数量等角度进行分类.通常情况下,对于计算机系统与计算机网络中的资源管理问题,由于资源的种类繁多,数量庞大,因而所建立的 MDP 模型通常会遇到"状态空间爆炸"问题,即 MDP 模型的状态空间随着问题规模指数级增长,这使得传统精确求解算法如值迭代与策略迭代等无法应用.因此,本文详细讨论了 MDP 模型的近似求解算法,将这些算法归为 3 类:贪心算法、基于状态聚合的算法以及基于近似动态规划(Approximate Dynamic Programming,ADP)的算法.

马尔可夫决策过程在实际应用中体现出一些不足,主要表现在:(1)模型不够直观.一方面,MDP 中的各个模型要素都使用了严格的形式化定义,虽然具有较强的逻辑性与严密性,但是模型的直观性与可理解性却相对较低.另一方面,模型建立需要较强的数学背景,例如在推导系统状态转移概率时,往往需要建模者具有一定的随机数学基础,增加了模型建立的难度.(2)在一些复杂应用环境中,单纯的 MDP 模型难以精确刻画系统的特点.例如,在网络安全问题中,MDP 难以描述网络拓扑与各个组件之间的逻辑关系.这些不足激励学者们进行了进一步的模型拓展研究,其中较有代表性的是马尔可夫决策 Petri 网(Markov Decision Petri Nets,MDPN)与随机博弈网(Stochastic Game Nets,SGN).这些模型方法将动态优化理论与随机 Petri 网理论相结合,在一定程度上克服了上述缺点.随机 Petri 网模型语义明确,使用图形化表示方式,直观易懂.系统中各个组件之间的关系可以灵活地使用组件之间的连接弧与变迁可实施函数等方式表现.建模者可以将精力更多地放在研究目标系统与精确描述系统与决策者行为方面,而状态转移概率等其它较为复杂的模型元素则可以利用 Petri 网工具中集成的功能实现自动化推导.

MDPN 模型将 MDP 理论与随机 Petri 网理论相结合,可以体现出系统与决策者宏观层面上的行为交替.利用 MDPN 模型,可以方便地借助 Petri 网图形工具对系统进行建模,并对模型的可达图进行规约得到 MDP 模型.SGN 模型是动态优化模型的进一步扩展,它将动态随机博弈与 Petri 网理论相结合,允许系统中存在多个决策者.每个决策者一般都有各自的目标函数,他们之间既可以是合作关系,也可以是竞争关系.在建立 SGN 模型时,可以先单独从各个决策者的角度出发,建立 SGN 子模型,再利用模型组合与化简技术,得到完整的 SGN 模型.求解 SGN 是一个寻求每个决策者均衡策略的问题,可归结为一个静态非线性规划问题.

动态优化模型是当前计算机系统与计算机网络的资源分配与任务调度等问题中的研究热点,对降低系统维护成本、提高系统运行效率具有重要的意义.本文从建模、求解与应用等角度,论述了马尔可夫决策过程、马尔可夫决策 Petri 网以及随机博弈网等动态优化模型在计算机系统与计算机网络中的应用.

2　基于马尔可夫决策过程的动态优化模型

2.1　马尔可夫决策过程

一个基本的马尔可夫决策过程包括以下要素：

(1) 状态集合 S，描述系统的状态.

(2) 行为集合 A，描述决策者在状态空间中可能的行为. 通常行为集合会依赖于当前状态，即可将其记为 $A(s)$.

(3) 收益函数 $R(s,s',a)$，$s,s'\in S$，$a\in A$，描述系统在决策者行为的影响下运行所产生的收益.

(4) 状态转移关系 S^M，描述系统状态在决策行为影响下的转移过程.

马尔可夫决策过程的一个显著特征是无后效性，即系统在下一时刻的状态仅依赖于当前所处的状态与决策行为，而与系统的历史无关.

根据 S^M 的性质不同，MDP 可以分为确定 MDP 与随机 MDP 两大类. 对于确定 MDP，在某个状态下的某个行为会导致唯一确定的状态转移，即 S^M：$S\times A\to S$，此时状态转移方程可记为 $s'=S^M(s,a)$；对于随机 MDP，未来系统状态不仅取决于当前系统状态下决策者的行为，还受到外部随机变量 W 的影响，即 S^M：$S\times A\times W\to S$，此时状态转移方程可记为 $s'=S^M(s,a,W(\omega))$，其中 $W(\omega)$ 为外部随机变量的一个实现样本. 随机 MDP 的未来状态一般服从某种分布，该分布可记为 $P(s'|s,a)$. 本文主要研究随机 MDP，下文中提到的马尔可夫决策过程，一般均指随机马尔可夫决策过程. 定义 $R(s,a)=\sum_{s'\in S}P(s'|s,a)R(s,s',a)$ 为状态 s 下采用行为 a 所产生的收益.

在马尔可夫决策过程中，策略 π 定义为从状态集合 S 到行为集合 A 的一个映射. 决策者根据策略 π 来得到当前所需的决策行为. 一个典型的马尔可夫决策过程的执行流程如下：

1. 决策者观察当前所处的状态 s.
2. 根据当前状态确定决策行为 $\pi(s)$.
3. 执行行为 $\pi(s)$，系统状态发生转换.
4. 重复 1.

MDP 在系统演进过程中，会产生一个收益序列. 为比较 MDP 中决策的优劣程度，引入了目标函数 J. 它将一个收益序列映射为一个单一的实数值. 对于无限时间 MDP 来说，其设置一般有 3 种方法：

(1) 在无限时间 MDP 中截取一个足够长的有限时间 MDP，则无限时间 MDP 的目标函数可近似地看作有限时间 MDP 的收益的和.

(2) 依照时间的推移对未来所得收益进行逐步折扣，保证对时间累加的总收益总是收敛的. 这种方式更看重当前所得的收益.

(3) 平均收益在时间趋于无穷处的极限值.

通过目标函数 J，可以定义策略之间的偏序关系，这样就可以对策略的优劣进行比较了.

MDP 中另一个重要概念是值函数 $V^\pi(s)$. $V^\pi(s)$ 是从 $\pi\times S$ 到实数集 $\mathbb{R}$ 的映射，其含义为在采用策略 π 的前提下，在状态 $s\in S$ 下所得到的目标函数 J 的期望. 无限时间 MDP 的值函数满足 Bellman 递推方程，即式(1)：

$$V^\pi(s_t)=R(s_t,\pi(s_t))+\alpha\sum_{s_{t+1}\in S}P(s_{t+1}|s,\pi(s))V^\pi(s_{t+1}) \tag{1}$$

其中 α 为折扣因子. 式(1)说明，给定策略 π，则在状态 s 的值函数等于当前一步决策所得收益与下一时刻折扣后值函数期望的和.

式(1)也可以写成如式(2)所示的向量形式，即

$$\boldsymbol{V}^\pi=\boldsymbol{R}^\pi+\alpha\cdot\boldsymbol{P}^\pi\boldsymbol{V}^\pi \tag{2}$$

2.2　马尔可夫决策过程建模与分析

在利用马尔可夫决策过程对系统进行建模分析时，可使用如下步骤：

(1) 明确系统运行目标.

该步骤中需要确定 MDP 的收益函数 R 与目标函数 J. 一方面，不同系统的运行目标可能不同. 另一方面，即使是对于同一系统，研究角度的差异也会导致不同的收益函数与目标函数. 以计算机网络为例，较为常用的目标函数有

① 节点吞吐量[1-3]；

② 能量消耗[4-6]；

③ 信道利用率[7-8]；

④ 延迟[9-10]；

⑤ 分组丢失率[11].

对于随机 MDP，通常使用带有期望形式(E)的目标函数. 一般期望目标函数具有如下形式：

有限马尔可夫决策过程：

$$J=\mathrm{E}\left\{\sum_{t=1}^{T}R(s_t,a_t)\right\} \tag{3}$$

无限马尔可夫决策过程：

$$J=\mathrm{E}\left\{\sum_{t=1}^{\infty}\alpha^tR(s_t,a_t)\right\} \tag{4}$$

$$J=\lim_{T\to\infty}\frac{1}{T}\mathrm{E}\left\{\sum_{t=1}^{T}R(s_t,a_t)\right\} \tag{5}$$

其中，s_t与a_t分别为t阶段系统所处状态与决策者采取的行为.式(4)与式(5)分别为无穷时间折扣情形与无穷时间平均情形下的目标函数.系统的运行目标通常是最大化或最小化上述目标函数J.

(2)确定系统运行状态空间与决策者的行为空间.

系统的状态空间与决策者的行为空间可能是离散可列的.例如，在认知无线电系统中，信道可以用两个离散的状态刻画，即{空闲，占用}，用户的行为也可能是离散的，如{发送数据，监听信道}.状态空间与行为空间也可能是连续的.例如，在上例中，若用户的行为变为“以概率p发送数据”，则用户行为空间是连续的，且其取值范围为[0,1].

(3)根据系统状态之间的动态转移关系建立Bellman递推方程.

该步骤中要找到状态之间的转移关系.对于随机MDP来说，转移关系包括状态转移方程$s'=S^M(s,a,W(\omega))$与转移概率$P(s'|s,a)$.有时状态转移概率无法精确得知，此时可以使用强化学习[12]的方法来求解马尔可夫决策过程.Bellman方程描述的是值函数V的递推关系，该方程在求解最优策略时发挥了重要作用.

(4)根据所建立的Bellman递推方程，对模型进行求解，得到最优策略π^*.

以最大化目标函数为例，求解过程中的关键步骤包括

$$\pi^*(s_t)=\arg\max_{a_t\in A}\left\{R(s_t,a_t)+\alpha\sum_{s_{t+1}\in S}P(s_{t+1}|s_t,a_t)V^*(s_{t+1})\right\} \tag{6}$$

$$V^*(s_t)=R(s_t,\pi^*(s_t))+\alpha\sum_{s'\in S}P(s_{t+1}|s_t,\pi^*(s_t))V^*(s_{t+1}) \tag{7}$$

式(6)按照最大化策略，根据当前所得的值函数V^*求得在状态s下应该采取的策略π^*，而式(7)则根据π^*计算其所对应的值函数.式(6)、(7)实际上是一个迭代的过程.不同的求解算法，如值迭代、策略迭代等，均需要使用以上两个步骤，只是顺序不同.

下面以一个接纳控制为例(图1)，举例说明MDP的建模方法.外部任务到达后，首先缓存在接纳控制器的等待队列中.在每个时间槽的开始，接纳控制器将其等待队列中的任务按照某种策略或将任务丢弃，或将任务分配给服务器1～n中的一个.在这个系统中，决策者为接纳控制器，其t时刻的行为是向量$\boldsymbol{x}_t=\{x_t^i\}$，其中每个分量$x_t^i$表示向服务器$i$分配的任务数.系统中的外部随机变量$W$包括两部分：①$[t-1,t]$内到达接纳控制器的任务数$\lambda_t$；②$[t,t+1]$内服务器$i$完成的任务数$\mu_t^i$.

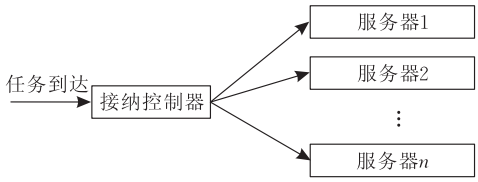


图1 MDP建模举例：一个接纳控制问题

系统在t时刻的状态可用$\{\lambda_t,q_t^1,\cdots,q_t^n\}$表示.其中$q_t^i$表示服务器$i$中的队列长度.假设若等待队列中的任务没有得到及时服务，则下一时刻这些任务会丢失.系统的状态转移方程可写为

$$q_{t+1}^i=q_t^i+x_t^i-\mu_t^i,\quad \forall i\in[1,\cdots,n] \tag{8}$$

此时决策行为受到如下流守恒条件约束：

$$\sum_{i=1}^{n}x_t^i\leq\lambda_t.$$

若假设系统每个时间槽内的收益与完成的任务数成正比，与服务器中驻留的任务数成反比，则系统在$[t,t+1]$时间段内的收益函数可定义如下：

$$r\sum_{i=1}^{n}\mu_i-\sum_{i=1}^{n}c_i(q_t^i+x_t^i),$$

其中r为每完成一个任务所得的收益，c_i为在服务器i上每个时间槽内服务一个任务所需要的成本.该系统的无穷时间折扣MDP的目标函数为

$$J=\mathrm{E}\left\{\sum_{t=1}^{\infty}\sum_{i=1}^{n}\alpha^t(r\mu_i-c_i(q_t^i+x_t^i))\right\}.$$

MDP的求解方法将在2.4小节进行详细讨论.

2.3 马尔可夫决策过程的分类

根据不同的划分依据，可将马尔可夫决策过程进行分类，如表1所示.

表1 马尔可夫决策过程的分类

划分依据	种 类
执行的时间	有限时间MDP，无限时间MDP
决策者的观测能力	完全可观测MDP，部分可观测MDP(POMDP)
转移关系的确定性	确定MDP，随机MDP
时间的连续性	连续时间MDP，离散时间MDP
转移概率/收益的确定性	普通MDP，带有强化学习的MDP
是否有附加限制条件	不受限MDP，受限MDP(CMDP)
目标的数量	单目标MDP，多目标MDP

(1)按照系统的执行时间分类.

现实中系统的运行时间都是有限的.对于有限时间马尔可夫决策过程，其目标函数可以简单地写成在系统运行期间内收益的和，如文献[3].当系统

运行时间很大时，也可以近似地认为系统的运行时间是无限的. 此时针对该系统建立的马尔可夫决策过程就是无限时间马尔可夫决策过程. 无限时间马尔可夫决策过程通常采用折扣累积收益或平均收益作为其目标函数，即式(4)和(5). Haas 等人[13]分别针对这两种目标函数研究了无线多媒体网络环境中的资源分配问题. 折扣累积收益目标函数已经被广泛研究，理论成果较为完善.

(2) 按照决策者的观测能力分类.

一般情形下，决策者可以完全观测到系统的状态，并根据所观测到的状态进行决策. 但是，有些时候决策者不能完全观测到系统状态. 这时，需要利用部分可观测马尔可夫决策过程(Partially Observable Markov Decision Process，POMDP)进行建模. Zhao 等人[2-3]研究了认知无线电系统中次用户的信道监听与接入的问题. 在该问题中，由于次用户监听可能发生错误，因此系统是一个部分可观测马尔可夫决策过程. POMDP 求解相较于 MDP 来说较为复杂. 因为决策者没有系统状态的精确信息，所以需要维护一个信任向量，用来描述系统当前位于各个状态的概率. 信任向量随着系统的演进而不断更新.

(3) 按照转移关系的确定性分类.

决策者在某个状态下所做的行为，有时会导致一个确定的结果，即以概率 1 转移到下一个状态，这称为确定马尔可夫决策过程. 有时，决策者的行为会导致不确定的结果，这称为随机马尔可夫决策过程.

(4) 按照时间的连续性分类.

现实中的一些问题是离散时间的，例如在库存管理问题中，仓库管理员一般每隔一个固定的时间间隔采购商品，更新库存. 还有一些问题是连续时间的，典型问题如队列管理问题[14]与设备维护问题[15-16]. 在这些问题中，系统的状态转换间隔时间(如顾客到达的间隔时间与设备正常运转的时间等)服从指数分布，每次系统状态发生转换时都需要决策者进行决策.

(5) 按照转移概率/收益的确定性分类.

在一些复杂系统中，系统的状态转移概率 $P(s'|s,a)$难以精确测量，收益函数 $R(s,a)$也无法推导出显式的解析式. 这时，就需要建立基于强化学习的 MDP 模型，采用跟踪实际系统运行过程或 Monte Carlo 模拟等方法，不断学习系统的未知特性. 值得注意的是，基于强化学习的 MDP 也是一种 MDP 的近似求解方法，简化了 Bellman 方程中值函数期望的计算.

(6) 根据是否有附加限制条件分类.

有时决策者行为会受到一些客观条件的影响，这时可将该问题归结为一个受限马尔可夫决策过程(Constrained Markov Decision Process，CMDP)问题. 以折扣情形为例，一个 CMDP 模型可表达为

$$\max_{a_t} \mathrm{E}\left\{\sum_{t=1}^{\infty}\alpha^t R(s_t,a_t)\right\}$$

$$\text{s.t.}: \mathrm{E}\left\{\sum_{t=1}^{\infty}\beta^t c(s_t,a_t)\right\}\leq C \tag{9}$$

约束(9)中 $c(s_t,a_t)$可视为阶段 t 所产生的资源消耗，C 为客观资源总量限制. 这类问题在计算机系统内有大量的应用，如 Djonin 等人[17]研究了在 MIMO 系统中，在数据延迟受限的情况下，最小化平均发射功率的问题. 求解 CMDP 可以使用线性规划法[18]与拉格朗日法[19]等.

(7) 按照目标数量分类.

很多动态优化问题只考虑一个目标函数，也就是常见的单目标优化问题. 如果目标函数有多个，就需要用多目标优化建模. 一般处理多目标问题的方式有 3 种：① 将一部分目标函数转化为约束，进而转化为 CMDP 模型[20]；② 将各个目标加权平均，组合成一个整体目标[21]；③ 求解帕雷托前沿(Pareto Frontier)[22-23].

2.4　马尔可夫决策过程的求解

由于篇幅所限，本文主要讨论无穷折扣马尔可夫决策过程的求解. 求解算法分类如图 2 所示.

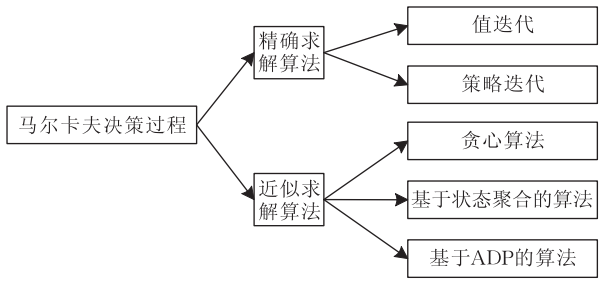


图 2　MDP 求解算法分类

2.4.1　精确求解算法

折扣情形下的最优解满足

$$\boldsymbol{V}^* = T(\boldsymbol{V}^*) \tag{10}$$

运算符 T 定义如下：

$$[T(\boldsymbol{V})]_s = \max_{a\in A(s)}\left\{R(s,a)+\alpha\sum_{s'\in S}P(s'|s,a)[\boldsymbol{V}]_{s'}\right\} \tag{11}$$

其中，$[\cdot]_s$为向量的第 s 个分量. 满足式(10)的值函数即是最优值函数 $\boldsymbol{V}^*$. 可以看到，式(11)实际上是采取最大化策略的式(1)的变形.

(1)值迭代算法.值迭代算法实际上是近似算法,随着迭代过程的进行,该算法会不断逼近最优解.值迭代算法如算法1所示.

算法1. 值迭代算法.

1. $n=0$,给定初值 $\boldsymbol{V}_0=\boldsymbol{v}$.
2. 根据迭代式 $\boldsymbol{V}_n=T(\boldsymbol{V}_{n-1})$,计算第 n 次迭代的值函数与策略.
3. 重复步2.

可以证明,算法1在 $n\to\infty$ 时收敛于最优值函数 $\boldsymbol{V}^*$.此外,还可以在每一次迭代时估计出最优解的区间,即

$$\boldsymbol{V}_n+\frac{\alpha}{1-\alpha}\cdot\beta_n\cdot\boldsymbol{e}\leq\boldsymbol{V}^*\leq\boldsymbol{V}_n+\frac{\alpha}{1-\alpha}\cdot\alpha_n\cdot\boldsymbol{e} \tag{12}$$

其中,$\boldsymbol{e}$ 为全1向量,α_n 与 β_n 定义如下:

$$\alpha_n=\max_{s\in S}\{[\boldsymbol{V}_n]_s-[\boldsymbol{V}_{n-1}]_s\} \tag{13}$$

$$\beta_n=\min_{s\in S}\{[\boldsymbol{V}_n]_s-[\boldsymbol{V}_{n-1}]_s\} \tag{14}$$

式(12)也可以作为值迭代算法运行结束的判定方法.例如,可事先指定一精度 ε,使得

$$\frac{\alpha}{1-\alpha}\cdot(\alpha_n-\beta_n)\cdot\boldsymbol{e}\leq\varepsilon$$

成立时算法终止.

(2)策略迭代算法.可以证明,当状态集合与行为集合有限时,策略迭代算法可以在有限迭代次数内获得最优解,且迭代次数上界为策略数,即 $\prod_{s\in S}|A(s)|$,其中 $|\cdot|$ 为集合内的元素个数.策略迭代算法如算法2所示.算法2首先确定一个初始策略 π_0,并直接根据式(2)求解得到该策略所对应的值函数.最后,再根据所得的值函数对策略进行更新.若更新前后的策略相同,则说明已经找到了最优策略,算法结束.

算法2. 策略迭代算法.

1. $n=0$,给定初始策略 π_0.
2. 通过求解 $(\boldsymbol{I}-\alpha\boldsymbol{P}^{\pi_n})\boldsymbol{V}_n=\boldsymbol{R}^{\pi_n}$ 确定 $\boldsymbol{V}_n$.
3. 确定 π_{n+1} 使其满足
$$\pi_{n+1}=\arg\max_{\pi_{n+1}}\{\boldsymbol{R}^{\pi_{n+1}}+\alpha\boldsymbol{P}^{\pi_{n+1}}\boldsymbol{V}_n\}.$$
4. if $\pi_{n+1}=\pi_n$,算法终止,设定最优策略 $\pi^*=\pi_n$.
else $n=n+1$,转到步2.

此外,学者们还基于以上两种基本算法设计了一些变形算法,如修正的策略迭代(Modified Policy Iteration)等,此处不再赘述.

2.4.2 近似求解算法

在一个实际系统中,资源种类与资源数量都极其庞大,导致所建立的MDP模型无法利用精确算法进行求解,原因在于:① 需要为每个状态存储其值函数.在状态数较多时,现有的技术无法提供足够的存储空间;② 在迭代过程中,计算值函数要遍历所有状态,会导致迭代一次所需时间较长,算法收敛速度太慢.基于这些考虑,人们开始寻找MDP的近似求解算法,使得在有限的时空复杂度范围内,得到可接受的次优解.

(1)贪心算法

贪心算法又称为近视策略(myopic policy),它可表示为:在时刻 t,求解如下优化问题

$$\max_{a_t\in A(s_t)}R(s_t,a_t) \tag{15}$$

例如,在图1的接纳控制问题中,贪心算法为

$$\max_{x_t^i}\sum_{i=1}^{n}\{r\mu_t^i-c_i(q_t^i+x_t^i)\}$$

$$\text{s.t.}:\sum_{i=1}^{n}x_t^i\leq\lambda_t.$$

贪心算法是最简单的一类近似算法.它只关注系统当前的收益,而忽略当前决策对未来收益的影响.这种方法虽然未必是最优的,但是至少提供了一种动态优化问题的简单求解方案.贪心算法的最大优点在于,其求解过程没有算法1与算法2中的迭代过程,因而时间复杂度较低.此外,也不需要提供存储值函数的空间.

在一些特殊的动态优化模型中,贪心策略就是最优策略.Karush与Dear[24]将一个学习过程利用POMDP建模,并证明了贪心策略在该类问题中的最优性.Krishnamurthy等人[25]研究了目标跟踪中的动态传感器调度问题,并给出了贪心策略是最优策略的一些充分条件.文献[26-28]分别从不同角度研究了机会频谱接入问题,并证明了贪心策略的最优性.

然而,通常情况下贪心策略并非最优策略.如Ahmad等人[29]指出,在负相关系统转移的机会频谱接入问题中,若信道都是独立同分布的Gilbert-Elliot信道,当信道数量大于3时,贪心策略并不是最优策略.虽然如此,在很多应用中,贪心算法表现出较好的适应性[30-31].

(2)基于状态聚合的算法

精确求解算法应用的最大障碍是状态空间爆炸问题.因此,一种很直观的近似求解策略是将问题空间进行聚合化简,使得问题规模减少,便于精确算法求解.

以图1中问题为例,假设接纳控制器的等待队列与所有服务器的服务队列最大容量均为100个任

务，则该问题 MDP 模型的状态空间共有 100^{n+1} 个状态. 此时，可设定如下状态聚合策略：为每个队列设定一个阈值，当队列长度低于该阈值时，则认为处于宏状态“低负载”，反之，则处于宏状态“高负载”. 这样，每个队列的状态可以化简为 2，整个系统的状态也缩小为 2^{n+1}.

一种常用的 MDP 状态聚合方法来源于马尔可夫过程的近似求解理论，见算法 3. 假设有状态转换如图 2 所示的马尔可夫决策过程. 如果存在一种对状态空间的划分，在每个划分内，任选一个状态，使得：① 以该状态作为起始状态，则转移到该状态所属划分内状态的概率很大；② 以该状态作为起始状态，则转移到不属于该状态所属划分内状态的概率很小，则这个 MDP 可以进行状态聚合化简. 例如在图 3 中，实线转移概率比虚线转移概率大很多，则该模型可以利用图中所示方式进行聚合. Liu 等人[32]利用该方法近似求解了分布式 Web 服务系统中的服务器选择问题.

算法 3.　MDP 状态空间化简算法.

1. 将状态空间 S 进行划分：$\{S_1, S_2, \cdots, S_n\}$.
2. for $i=1$ to n
3.　将所有转到 S_i 以外状态的概率都设置为 0.
4.　将 S_i 内的状态转移概率进行归一化处理.
5.　计算 S_i 内状态的稳态概率分布，利用 $\pi=\pi \boldsymbol{P}$，其中 $\boldsymbol{P}$ 为归一化的 S_i 内部转状态移概率矩阵.
6.　计算 S_i 到 S_j 的转移概率 $P_{ij}=\sum_{k\in S_i}\pi_k p_{kj}$.
7. end for

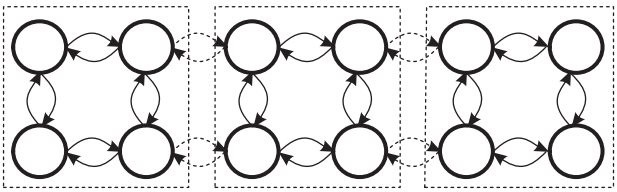


图 3　一个 MDP 状态空间的聚合

值得注意的是，只有在上面两个假设条件都满足的时候，算法 3 才能得到精度较高的近似解. 否则，误差会比较大. 此外，这种近似方法还有一个缺点，即不能得到近似解与精确解之间的关系. 为了克服这些问题，学者们又提出了其它解决方案，有界参数 MDP（Bounded-Parameters MDP，BMDP）就是这些方案中较有影响力的方法之一.

BMDP 由 Givan 等人[33-34]提出，它是非精确状态转移概率 MDP（Markov Decision Processes with Imprecisely Known Transition Probabilities，MDPIPs）模型的一种特殊情况. BMDP 是一个 4 元组 $\{S, A, R_\updownarrow, P_\updownarrow\}$. 与传统 MDP 不同，在 BMDP 中每个状态的收益函数 $R_\updownarrow$ 与状态转移概率 $P_\updownarrow$ 是一个区间，而不是一个点值. 若一个 MDP M，其状态、行为集与 BMDP $M_\updownarrow$ 的状态、行为集完全相同，且 M 的收益函数 R 与转移概率 P 都在 $M_\updownarrow$ 所规定的区间内，则称 $M\in M_\updownarrow$.

在一个 BMDP $M_\updownarrow$ 中，给定一个决策策略 π，则该策略所产生的值函数也是一个区间，称为区间值函数

$$V_\updownarrow^\pi(s)=\left[\min_{M\in M_\updownarrow} V_M^\pi(s), \max_{M\in M_\updownarrow} V_M^\pi(s)\right] \quad (16)$$

其中 $V_M^\pi(s)=R_M(s,a)+\alpha\sum_{s'\in S}P_M(s'|s,a)V_M^\pi(s')$ 是 $M_\updownarrow$ 中的一个 MDP M 的值函数. 可根据实际工程应用背景，定义区间值函数的比较方法. 例如对于策略 a 与 b，可定义：

① 乐观最优

$$V_\updownarrow^a \gg V_\updownarrow^b \Leftrightarrow V_\uparrow^a \geq V_\uparrow^b \vee (V_\uparrow^a = V_\uparrow^b \wedge V_\downarrow^a \geq V_\downarrow^b).$$

② 悲观最优

$$V_\updownarrow^a \gg V_\updownarrow^b \Leftrightarrow V_\downarrow^a \geq V_\downarrow^b \vee (V_\downarrow^a = V_\downarrow^b \wedge V_\uparrow^a \geq V_\uparrow^b).$$

可以证明，存在 $M\in M_\updownarrow$，使得所有状态的值函数能同时达到最大或最小，并称这两个 MDP 分别为关于策略 π 的最大 MDP 与最小 MDP. 寻找最大或最小 MDP 的过程，相当于寻找关于值函数上界降序排列状态空间序列与值函数下界升序排列状态空间序列的序列最大 MDP（Order Maximizing MDP）. 具体来讲，一个状态空间序列 $O=\{s_1, s_2, \cdots, s_n\}$ 为状态空间中所有状态的一个排列顺序，则状态空间序列 O 的序列最大下标 r 与序列最大 MDP M_O 可定义如下.

定义 1（序列最大下标与序列最大 MDP）[34]. 对于某个状态 s 与决策行为 a，其关于序列 O 的序列最大下标 r 为

$$\arg\max_{1\leq r\leq n}\sum_{i=1}^{r-1}P_\uparrow(s_i|s,a)+\sum_{i=r}^{n}P_\downarrow(s_i|s,a) \quad (17)$$

相应的序列最大 MDP 是一个满足式(18)的 MDP $M_O\in M$

$$P_{M_O}(s_i|s,a)=\begin{cases}P_\uparrow(s_i|s,a), & i<r\\ P_\downarrow(s_i|s,a), & i>r\end{cases} \quad (18)$$

$$P_{M_O}(s_r|s,a)=1-\sum_{i=1,i\neq r}^{n}P_{M_O}(s_i|s,a) \quad (19)$$

利用 BMDP 可以对问题空间进行状态聚合. 一个精确 MDP 经过聚合后，一般都可以归结为一个 BMDP 问题，可以利用区间迭代求解算法进行求解，即

$$IVI_{\updownarrow\mathrm{opt}}(V_{\updownarrow})(s)=\max_{a\in A(s)}\left[\min_{M\in M_{\updownarrow}}VI^{a}_{M}(V_{\downarrow})(s),\max_{M\in M_{\updownarrow}}VI^{a}_{M}(V_{\uparrow})(s)\right] \quad (20)$$

计算式(20)实际上可以看作为具有2个决策者的2步博弈过程.以乐观最优为例,在第1步中,决策者1与决策者2为合作配合关系,决策者1利用所定义的乐观最优比较运算符≥求得最大化区间值函数上界的策略$\pi_{\uparrow\mathrm{opt}}$.在第2步中,决策者1与决策者2为对立竞争关系,决策者2求得策略$\pi_{\uparrow\mathrm{opt}}$的最小MDP,并计算区间值函数的下界.该过程可用算法4描述.其中,*Sort_Dec_Order*与*Sort_Inc_Order*为排序函数,*Order_Max_Ind*利用式(17)求得对应的序列最大下标.这样,就可以在缩小的问题空间中,求得原问题具有边界的解.

算法4. 区间迭代算法.

1. $O_{\mathrm{up}}=Sort\_Dec\_Order(V_{\uparrow})$,
   $O_{\mathrm{down}}=Sort\_Inc\_Order(V_{\downarrow})$.
2. for all $s\in S$ do
3. for all $s\in S$ do
4. $r_{\mathrm{up}}=Order\_Max\_Ind(M_{\updownarrow},O_{\mathrm{up}},s,a)$.
5. $r_{\mathrm{down}}=Order\_Max\_Ind(M_{\updownarrow},O_{\mathrm{down}},s,a)$.
6. for $i=1$ to n do
7. 根据式(18)、(19)计算$P_{\mathrm{up}}(s_{O_{\mathrm{down}}(i)}|s,a)$与$P_{\mathrm{down}}(s_{O_{\mathrm{up}}(i)}|s,a)$.
8. end for
9. end for
10. $V_{\uparrow}=\max_{\boldsymbol{a}\subseteq A(s)}R_{\uparrow}(s,a)+\alpha\sum_{s'\in S}P_{\mathrm{up}}(s'|s,a)V_{\uparrow}(s')$.
11. if $|a|=1$ and $a=\{a\}$ then
12. $V_{\downarrow}=R_{\downarrow}(s,a)+\alpha\sum_{s'\in S}P_{\mathrm{down}}(s'|s,a)V_{\downarrow}(s')$.
13. $\pi(s)=a$.
14. else
    $V_{\downarrow}=\max_{a\in\boldsymbol{a}}R_{\downarrow}(s,a)+\alpha\sum_{s'\in S}P_{\mathrm{down}}(s'|s,a)V_{\downarrow}(s')$.
15. $\pi(s)=a$.
16. end if
17. end for

(3)基于近似动态规划的算法

近似动态规划(Approximate Dynamic Programming,ADP)是一种解决大规模动态优化问题的现代近似求解方法.目前,关于近似动态规划的代表性专著,主要有3本[12,35-36],分别从人工智能、控制论以及运筹学的角度对近似动态规划进行了详细的论述.近似动态规划能有效解决马尔可夫决策过程中的状态空间爆炸问题.

在ADP中,式(11)通常改写为

$$V(s_t)=\max_{a_t\in A(s_t)}R(s_t,a_t)+\alpha\cdot\mathrm{E}\{V(s_{t+1})\} \quad (21)$$

在式(21)中,状态空间爆炸问题表现为:(1)问题状态空间S太大,现有的技术无法提供足够的存储空间;(2)外部随机变量有时无法精确测量其分布,或即使分布已知,也会由于随机变量状态太多而导致其期望难于计算.在近似动态规划中,主要使用基于值函数近似(Value Function Approximation)与后决策状态(Post-Decision State Variable)的前向动态规划方法来克服以上问题.

令系统的状态转换方程为

$$s_{t+1}=S^{M}(s_t,a_t,W(\omega_t)) \quad (22)$$

其中,$W(\omega_t)$是t时刻外部随机变量的一个样本,则基本的近似动态规划算法可表述为算法5.

算法5. 基本近似动态规划算法.

1. 初始化:
   对每个状态s,初始化$\bar{V}(s)$,
   选择初始状态s_0.
2. for $t=0$ to T do
3. 求解
   $\hat{v}_t=\max_{a_t\in A(s_t)}\{R(s_t,a_t)+\alpha\cdot\mathrm{E}\{\bar{V}(s_{t+1})|s_t\}\}$,
   并令a_t为以上最大化问题的解.
4. 利用下式对$\bar{V}(s_t)$进行更新
   $\bar{V}(s_t)\leftarrow(1-\eta_t)\bar{V}(s_t)+\eta_t\hat{v}_t$.
5. 选定一个采样路径ω_t.
6. 计算下一个状态
   $s_{t+1}=S^{M}(s_t,a_t,W(\omega_t))$.
7. end for

算法5首先初始化所有状态的值函数,并指定一个初始状态.然后,利用Monte Carlo方法对随机外部信息进行一次采样.算法的核心是步2~7,首先求解一步优化问题(步3),并利用所得出的$\hat{v}_t$对值函数进行更新.其中η_t是步长.

该算法与用于求解一般马尔可夫决策过程迭代算法的最根本区别,在于时间是顺序演进的,而不是倒序演进的.算法运行的过程,实际上是一个系统仿真的过程.以图1中接纳控制问题为例,其ADP求解算法可描述如下:

1. 设定每个状态值函数的初始值,选取起始状态,并令$t=0$.

2. 采集t时刻系统状态,根据当前的值函数,计算当前决策行为x_t,并得到当前状态值函数的一个样本$\hat{v}_t$(算法5步3).其中,$\mathrm{E}\{\bar{V}(s_{t+1})|s_t\}$可用Monte Carlo模拟的方法求得.

3. 根据值函数样本,更新当前状态的值函数(算法5步4).

4. 使用 Monte Carlo 方法得到外部随机变量的样本，即任务到达数 λ_t 与任务完成数 μ_t^i.

5. $t \leftarrow t+1$，并根据式(8)得到 $t+1$ 时刻的系统状态，重复步 2.

该算法其优点显而易见，在迭代过程中不需要枚举系统的所有状态来计算值函数，一定程度上规避了状态空间爆炸问题.

但是，算法 5 中仍然存在不足. 例如，该算法为每个状态均设立一个变量 $\bar{V}(s)$ 用以存储其值函数. 当问题状态空间较大时，难以提供足够的存储空间. 同时，该算法只更新所遍历到的状态的值函数，而未遍历到的状态的值函数却得不到更新. 下面我们就算法 5 中的各个步骤展开论述，详细介绍近似动态规划算法克服状态空间爆炸问题的主要手段.

① 后决策状态

后决策状态是决策者做完决策后、且外部随机信息到达前系统的状态. 这样，式(22)就分为了两步：

$$s_t^x = S^{M,x}(s_t, a_t) \tag{23}$$

$$s_{t+1} = S^{M,\omega}(s_t^x, W(\omega_t)) \tag{24}$$

其中，s_t^x 称为 t 时刻的后决策状态，s_{t+1} 称为 $t+1$ 时刻的前决策状态. 后决策状态可以看作为前决策状态与决策行为的确定函数.

以图 1 中接纳控制问题为例，$\{\lambda_t, q_t^1, \cdots, q_t^n\}$ 为系统的前决策状态，而其后决策状态为 $\{q_t^{x,1}, \cdots, q_t^{x,n}\}$，它们之间的状态转移如下

$$q_t^{x,i} = q_t^i + x_t^i,$$

$$q_{t+1}^i = q_t^{x,i} - \mu_t^i.$$

后决策状态的值函数定义如下：

$$V(s_t^x) = \mathrm{E}\{V(s_{t+1}) \mid s_t^x\} \tag{25}$$

即它是下一时刻前决策状态值函数的期望. 此时，步 3 可以改写为

$$\hat{v}_t = \max_{a_t \in A(s_t)} R(s_t, a_t) + \alpha \cdot \bar{V}(S^{M,x}(s_t, a_t)) \tag{26}$$

注意到式(26)中，等式右边已经没有期望运算.

② 值函数近似

在算法 1 与算法 2 中，值函数表现为一种“查表”形式(Table Lookup Form)，即算法需要维护一个值函数表，表项为每个状态 s 所对应的值函数 $V(s)$. 这种方式使得值函数的存储与计算都较为困难. 在 ADP 中，可以利用函数近似的方法，利用一些简单的函数形式拟合后决策状态的值函数. 线性值函数近似是普遍使用的一种值函数近似方法. 令 $\mathcal{F}$ 为动态优化问题中的特征集，该特征集与问题结构本身有较大相关性. 如在分布式库存管理问题中，特征集可以包括各地库存量、各地仓库在单位时间内到达的货物量、库存变化的方差以及这些特征的平方等[37].

定义基函数(Basis Function) $\phi_f(s_t^x)$，$f \in \mathcal{F}$ 为关于后决策状态 s_t^x 中某一特征 f 数量关系的函数，即 $\phi_f(s_t^x)$ 为从后决策状态集合到实数集合的映射，则后决策状态的值函数可以利用如下方式进行近似：

$$V(s_t^x) \approx \bar{V}(s_t^x \mid \boldsymbol{\theta}) = \sum_{f \in \mathcal{F}} \theta_f \phi_f(s_t^x) \tag{27}$$

此时算法 5 中步 3 可以进一步改写为

$$\hat{v}_t = \max_{a_t \in A(s_t)} \{R(s_t, a_t) + \alpha \cdot \sum_{f \in \mathcal{F}} \theta_f \phi_f(s_t^x)\} \tag{28}$$

这样，估计值函数的过程，就转化为估计 θ_f 的过程，即 θ_f 随时间演进而不断更新，因此也可记作 $\theta_{f,t}$. 一般情况下，特征集的空间远小于问题的状态空间. 因此，值函数近似可以较好地解决状态空间爆炸的问题.

③ 值函数样本的取得

2.1 小节提到，状态 s_t 的值函数 $V(s_t)$ 为从状态 s_t 开始到时间趋于无穷时收益函数的累加. 在策略 π 作用下，$V(s_t)$ 的一个无偏估计样本可以直观地写为

$$\hat{v}(s_t) = \sum_{\tau=t}^{\infty} \alpha^{\tau-t} R(s_\tau, a_\tau^\pi) \tag{29}$$

式(29)可以用一个有限时间累计收益进行近似，即取一个足够大的 T，使得 $\alpha^{T-t} \to 0$，则

$$\hat{v}(s_t) \approx \sum_{\tau=t}^{T} \alpha^{\tau-t} R(s_\tau, a_\tau^\pi) \tag{30}$$

式(29)还可改写为

$$\hat{v}(s_t) = \sum_{\tau=t}^{\infty} \alpha^{\tau-t} R(s_\tau, a_\tau^\pi) - \sum_{\tau=t}^{\infty} \alpha^{\tau-t} (V(s_\tau) - \alpha V(s_{\tau+1})) + V(s_t) - \alpha^\infty V(s_\infty) \tag{31}$$

由于 $\alpha \in (0,1)$ 且 $V(s)$ 有界，因而 $\alpha^\infty V(s_\infty) \to 0$，式(31)可近似地变换为

$$\hat{v}(s_t) = V(s_t) + \sum_{\tau=t}^{\infty} \alpha^{\tau-t} (R(s_\tau, a_\tau^\pi) - V(s_\tau) + \alpha V(s_{\tau+1})) \tag{32}$$

其中 $R(s_\tau, a_\tau^\pi) - V(s_\tau) + \alpha V(s_{\tau+1})$ 称为即时差分(Temporal Difference，TD)或 Bellman 误差(Bellman Error)，表示当前值函数估计值与上次值函数估计值之间的差. 在一些文献中，折扣因子 α 有时用 λ 表示，因此这种取得值函数样本的方法又叫做 $TD(\lambda)$. 当折扣因子 $\alpha = 0$ 时，又可以得到一种特殊的表示方式：

$$\hat{v}(s_t) = R(s_t, a_\tau^\pi) + \alpha V(s_{t+1}) \tag{33}$$

式(33)称为 $TD(0)$. 注意式(33)与带有后决策状

态变量的 Bellman 方程(26)极为相似. 当 π 为式(26)中的最大化策略时,$V(s_{t+1})$ 为式(26)中 $\bar{V}(S^{M,x}(s_t,a_t))$的无偏样本.

当利用形如式(27)所示的值函数近似方法时,ADP 算法并不关注值函数本身,而着重考察值函数的导数 θ_f. 例如,在资源管理问题中,$\phi_f(s_t^x)$可以代表具有某一特性 f 的资源的数量,这时,θ_f的物理含义是该类资源的边际收益[38]. θ_f的样本可以通过以下两种方法得到:

(i) 优化问题的对偶变量. 一般资源管理问题都存在资源数量的约束,该约束所对应的对偶变量 $\hat{\theta}_f$就是资源的影子价格,即 θ_f的样本.

(ii) 数值微分. 在状态 s_t,根据式(33)可得 $\hat{v}(s_t)$. 此时,可将状态 s_t的 f 类资源的数量减 1 得到状态 $\tilde{s}_t$,重新进行优化,得到 $\hat{v}(\tilde{s}_t)$,则数值微分可表示为

$$\hat{\theta}_f=\hat{v}(s_t)-\hat{v}(\tilde{s}_t).$$

④ 值函数更新方法

随机梯度法是一种常用的值函数更新方法,可以通过逐步学习值函数样本 $\hat{v}$,使 $\bar{V}$ 不断逼近真实值函数. 随机梯度法的目标是

$$\min_{\bar{V}(s)} \mathrm{E}\left\{\frac{1}{2}(\bar{V}(s)-\hat{v}(s))^2\right\} \tag{34}$$

即寻找最符合样本 $\hat{v}$ 的值函数 $\bar{V}$. 由于 $\hat{v}(s)$是一个随机变量,因此该问题为一个随机优化问题,其求解算法与静态优化问题中的梯度法类似,称为随机梯度法. 若在 t 时刻,系统位于状态 s,对应的步长为 η_t,则

$$\bar{V}(s)\leftarrow\bar{V}(s)-\eta_t(\bar{V}(s)-\hat{v}(s))=(1-\eta_t)\bar{V}(s)+\eta_t\hat{v}(s).$$

注意到该式就是算法 5 中步 4.

若使用后决策状态的值函数,则优化目标(34)变为

$$\min_{\bar{V}(s_t^x)}\mathrm{E}\left\{\frac{1}{2}(\bar{V}(s_{t-1}^x)-\hat{v}(s_t))^2\right\},$$

此时更新方法为

$$\begin{aligned}\bar{V}(s_{t-1}^x)&\leftarrow\bar{V}(s_{t-1}^x)-\eta_t(\bar{V}(s_{t-1}^x)-\hat{v}(s_t))\\&=(1-\eta_t)\bar{V}(s_{t-1}^x)+\eta_t\hat{v}(s_t)\end{aligned} \tag{35}$$

若使用形如式(27)的后决策状态值函数近似策略,则随机梯度法的目标变为

$$\min_{\boldsymbol{\theta}}\mathrm{E}\left\{\frac{1}{2}(\bar{V}(s_{t-1}^x|\boldsymbol{\theta})-\hat{v}(s_t))^2\right\} \tag{36}$$

即寻找最接近实际值函数的后决策状态近似值函数 $\bar{V}(s_{t-1}^x|\boldsymbol{\theta})$. 此时只需更新 $\boldsymbol{\theta}$:

$$\boldsymbol{\theta}\leftarrow\boldsymbol{\theta}-\eta_t(\bar{V}(s_{t-1}^x|\boldsymbol{\theta})-\hat{v}(s_t))\nabla_{\boldsymbol{\theta}}\bar{V}(s_{t-1}^x|\boldsymbol{\theta}) \tag{37}$$

其中,由式(27)得

$$\nabla_{\boldsymbol{\theta}}\bar{V}(s_{t-1}^x|\boldsymbol{\theta})=\begin{pmatrix}\dfrac{\partial\bar{V}(s_{t-1}^x|\boldsymbol{\theta})}{\partial\theta_1}\\[2ex]\dfrac{\partial\bar{V}(s_{t-1}^x|\boldsymbol{\theta})}{\partial\theta_2}\\[2ex]\vdots\\[2ex]\dfrac{\partial\bar{V}(s_{t-1}^x|\boldsymbol{\theta})}{\partial\theta_{|\mathcal{F}|}}\end{pmatrix}=\begin{pmatrix}\phi_1(s_{t-1}^x)\\\phi_2(s_{t-1}^x)\\\vdots\\\phi_{|\mathcal{F}|}(s_{t-1}^x)\end{pmatrix} \tag{38}$$

此外,还有一些基于线性回归的值函数更新算法,如最小二乘即时差分(Least Squares Temporal Differences,LSTD)与最小二乘策略估计(Least Squares Policy Evaluation,LSPE)[35]. 这两种算法的主要区别在于,LSTD 采集所有值函数样本后一次进行拟合,而 LSPE 为一种边采集值函数样本边拟合的递归算法.

⑤ 状态聚合

2.4.2 节介绍了一些基于状态聚合的近似求解算法. 事实上,ADP 中也可以使用状态聚合. 不失一般性,一个聚合状态 s_g的值函数 $V(s_g)$可定义为该聚合状态所包含状态的值函数的平均值,即

$$V(s_g)=\frac{\sum_{s\in s_g}V(s)}{|s_g|} \tag{39}$$

状态聚合解决了状态空间爆炸问题,但是随之而来的问题是如何确定合理的状态聚合策略以获得较好的近似解. George 等人[39]提出了一种多层状态聚合的思想,将状态的近似值函数定义为不同层次聚合值函数的加权平均. 令 G 为聚合层次的集合,则

$$V(s)=\sum_{g\in G}w_gV(s_g) \tag{40}$$

其中 s_g为非聚合状态 s 在第 g 层聚合中所对应的聚合状态. w_g可以通过跟踪各层聚合状态值函数的误差与方差等参数确定. 这种方法在实际样本较少的问题中,显示出较强的适应性,可以加速算法的收敛速度.

⑥ 步长

步长一般可分为两类,一类是确定步长,如 $\eta_t=1/(t+1)$或 $\eta_t=a/(t+a)$等;另一类是随机步长,这类步长与每次取得的样本 $\hat{v}$ 或 $\hat{\theta}$ 等相关,一般收敛速度较快. 本文中以确定步长为例,简要介绍 ADP 值函数更新算法中的步长.

为保证随机梯度法收敛,一般要求确定步长 η 满足如下条件:(i) $\eta_t\geq0$;(ii) $\sum_{t=0}^{\infty}\eta_t=\infty$;(iii) $\sum_{t=0}^{\infty}\eta_t^2<$

∞. 在算法 5 的步 4 中，由于值函数样本 $\hat{v}_t$ 与所估计的值函数 $\bar{V}(s_t)$ 单位相同，因而可以简单地取 $0\leq\eta_t\leq 1$，如令 $\eta_t=1/(t+1)$，$t=0,1,\cdots$. 然而，在随机梯度法式(37)中，由于等式右边 $(\bar{V}(s|\boldsymbol{\theta}_{t-1})-\hat{v})\nabla_{\boldsymbol{\theta}}\bar{V}(s|\boldsymbol{\theta}_{t-1})$ 与 $\boldsymbol{\theta}$ 的单位不一定相同，η 的取值还需仔细调整. Powell 对步长进行了较为详细的介绍[36]，有兴趣的读者可以参考.

⑦ 探索(Exploration)与利用(Exploitation)问题

算法 5 中采用前向动态规划法，且下一个状态 s_{t+1} 的选取都与当前状态 s_t 所做的决策 a_t 有关，这称为依照策略的学习方式(On-Policy Learning). 这种方法充分利用了前期估计得到的统计信息，会不断提高所遍历到的状态的值函数，而没有遍历过的状态的值函数的数值则相对较低. 这很容易导致算法收敛于局部最优解而非全局最优解.

针对这个问题，学者们又提出了不依照策略的学习方式(Off-Policy Learning). 但是，这种方式不能保证 ADP 算法收敛. 因此，又提出了一些折中的方案，如 Boltzmann 探索[40]等，在算法前期，先利用 Off-Policy Learning 遍历尽量多的状态，采集足够多的统计信息，而在算法后期，则使用 On-Policy Learning 方法，加快收敛速度.

3　基于马尔可夫决策 Petri 网的动态优化模型

Beccuti 等人[41-42]于 2007 年提出了马尔可夫决策 Petri 网(Markov Decision Petri Nets，MDPN)，将 MDP 的思想融入了 Petri 网中，其目的是为了提供一种比 MDP 更高层的建模工具，从宏观的角度反映决策者行为与系统行为的交替，并从语义的角度严格定义两种行为的转换过程.

3.1　马尔可夫决策 Petri 网

马尔可夫决策 Petri 网可分为两种子网：代表系统行为的随机子网(Probabilistic Subnet)以及代表决策者行为的非确定子网(Nondeterministic Subnet). 这两种子网通过立即变迁 $NdtoPr$ 与 $PrtoNd$ 同步. 随机子网的行为通过两类变迁 $Trun^{pr}$ 与 $Tstop^{pr}$ 来描述. $Trun^{pr}$ 代表系统运行的中间过程，而 $Tstop^{pr}$ 代表系统当前阶段运行过程的终止. 随机子网中的每个变迁都对应一个权值(weight)，用来计算某个状态下系统可实施变迁的概率. 此外，每个变迁还对应系统中一个触发该变迁的行为(act)，包括组件集合的一个子集.

在 MDPN 中，系统由多个组件构成. 这些组件有些是可控的，有些是不可控的. 非确定子网用两类变迁 T_g^{nd} 与 T_l^{nd} 来描述. T_g^{nd} 代表决策者系统级的控制行为，而 T_l^{nd} 代表组件级的控制行为. 与随机子网中的变迁类似，非确定子网中的这两类变迁又可以细分为 $Trun_g^{nd}$、$Tstop_g^{nd}$、$Trun_l^{nd}$、$Tstop_l^{nd}$. 每个非确定子网中的变迁还对应一个对象，用以说明该变迁对应行为的施加组件对象.

定义 2(马尔可夫决策 Petri 网)[42]. 一个马尔可夫决策 Petri 网是一个四元组 $MN=\{Comp^{pr},Comp^{nd},N^{pr},N^{nd}\}$，其中

$Comp^{pr}$ 是一个有限非空系统组件集合；

$Comp^{pr}\subseteq Comp^{pr}\cup\{id_s\}$ 是非空可控组件集合，其中 id_s 代表整个系统；

N^{pr} 由 3 部分构成：① 一个带有优先级的 Petri 网 $\{P,T^{pr},I^{pr},O^{pr},H^{pr},prio^{pr},m_0\}$；② 一个对应的权值 $weight$：$T^{pr}\to\mathbb{R}$；③ 一个对应的行为 act：$T^{pr}\to 2^{Comp^{pr}}$，其中 $T^{pr}=Trun^{pr}\cup Tstop^{pr}$；

N^{nd} 由两部分构成：① 一个带有优先级的 Petri 网 $\{P,T^{nd},I^{nd},O^{nd},H^{nd},prio^{nd},m_0\}$；② 一个对应的对象 obj：$T^{nd}\to Comp^{nd}$，其中 $T^{nd}=Trun^{nd}\cup Tstop^{nd}$.

此外，MDPN 还需满足以下条件：① 一个变迁不能既是非确定变迁又是随机变迁；② 每个系统组件至少可以触发一个 $Tstop^{pr}$ 类型的变迁；③ 每个可控系统组件至少是一个 $Tstop^{nd}$ 类型变迁的对象.

在 MDPN 中，收益分为两部分. 第一部分是状态收益，即系统到达某个状态后得到的收益. 第二部分是行为收益，定义为一连串决策行为所得到的收益. 行为收益与行为序列的顺序无关.

3.2　马尔可夫决策 Petri 网的建模与分析

当构建好决策者行为子模型与系统行为子模型后，需要加入一些附加的位置与变迁，将两个子模型连接起来.

一个基本的 MDPN 模型如图 4 所示. 位置 $Stop_i^{pr}$、Run_i^{pr}、$Stop_i^{nd}$、Run_i^{nd}、$Stop_0^{nd}$、Run_0^{nd} 用来在系统组件、整个系统以及决策者之间进行同步. 对于每个组件 i，都有一个 $Stop_i^{pr}$ 与 Run_i^{pr} 位置. 若决策者采取了针对整个系统的全局性行为，则需插入位置 $Stop_0^{nd}$ 与 Run_0^{nd}. 若采取的是针对某个系统组件的局部行为，则需插入位置 $Stop_i^{nd}$ 与 Run_i^{nd}.

变迁 $NdtoPr$ 与 $PrtoNd$ 描述系统行为与决策者行为的交替进行. $NdtoPr$ 只有在 $Stop_0^{nd}$ 与所有

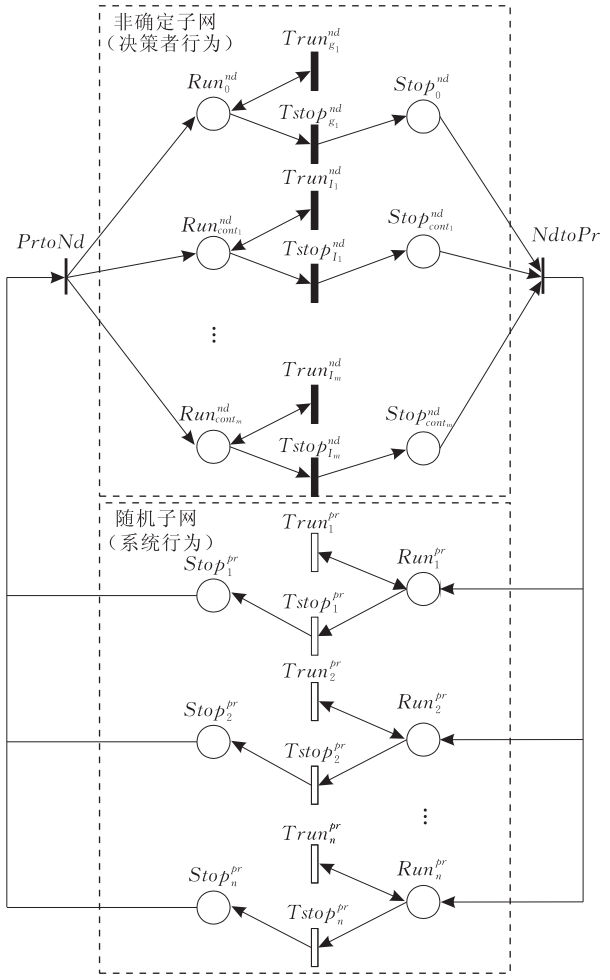


图 4 一个基本的 MDPN 模型

$Stop_i^{nd}$ 位置都有标记时才能实施,代表模型由决策态转移到系统运行态. 而 $PrtoNd$ 相反只有在 $Stop_i^{pr}$ 位置都有标记才能实施,代表由系统运行态转移到决策态.

3.3 马尔可夫决策 Petri 网的求解

MDPN 的求解过程可以分为如下 4 个步骤[41]:

(1) 由 MDWN 模型求得该模型的可达图 RG

可达状态集合(Reachability Set,RS)可分为两部分:非确定状态(RS_{nd})与随机状态(RS_{pr}). 在非确定状态中,只有 T_{nd} 类型的变迁是可实施的,而在随机状态中,只有 T_{pr} 类型的变迁是可实施的.

(2) 将可达图 RG 规约为非确定可达图 RG_{nd}

在 RG 中,定义非确定子路径与随机子路径分别为 RG 中经过同样类型状态的最大路径. 搜索所有非确定子路径,并将每个非确定子路径压缩为一个决策状态,代表所有可能的决策行为,得到非确定可达图 RG_{nd}.

(3) 将非确定可达图 RG_{nd} 规约为 MDP 可达图 RGMDP

搜索所有随机子路径,通过路径途中经过变迁的权值,计算各个路径的概率,并将每个随机子路径压缩为 RG 中的一条有向弧,代表宏观的系统状态转移,得到 MDP 可达图 RG_{MDP}.

(4) 计算对应 MDP 的转移概率

转移概率矩阵为

$$\boldsymbol{P}=\left(\sum_{n=0}^{\infty}(\boldsymbol{P}^{(pr,pr)})^n\boldsymbol{P}^{(pr,nd)}\right) \tag{41}$$

其中,$\boldsymbol{P}^{(pr,pr)}$ 为 RG 中从一个随机状态转移到另一个随机状态、且途中没有非确定状态的概率,$\boldsymbol{P}^{(pr,nd)}$ 为从一个随机状态转移到非确定状态的概率. 转移矩阵 $\boldsymbol{P}$ 可用式(42)进行计算:

$$\boldsymbol{P}=\begin{cases}\left(\sum_{k=0}^{n_0}(\boldsymbol{P}^{(pr,pr)})^k\right)\boldsymbol{P}^{(pr,nd)}, & \text{若随机状态集合不存在回路}\\ (\boldsymbol{I}-\boldsymbol{P}^{(pr,pr)})^{-1}\boldsymbol{P}^{(pr,nd)}, & \text{若随机状态集合中存在回路}\end{cases} \tag{42}$$

(5) 根据 Bellman 方程计算 MDP 中的最优策略

可根据算法 1 或算法 2 求得 MDP 中的最优策略,也就是 MDWN 模型中的最优控制策略.

3.4 应用与扩展

本小节中将以一个可修复系统为例[41],对 MDWN 模型的各个要素进行说明,其模型如图 5 所示. 左半部分为随机子网,描述一个既可能正常工作(变迁 $Workfine$)、又可能失效(变迁 $FailProc$)的系统组件. 右半部分为非确定子网,描述决策者的行为,包括分配资源以维修失效组件(变迁 $AssignRes$)与

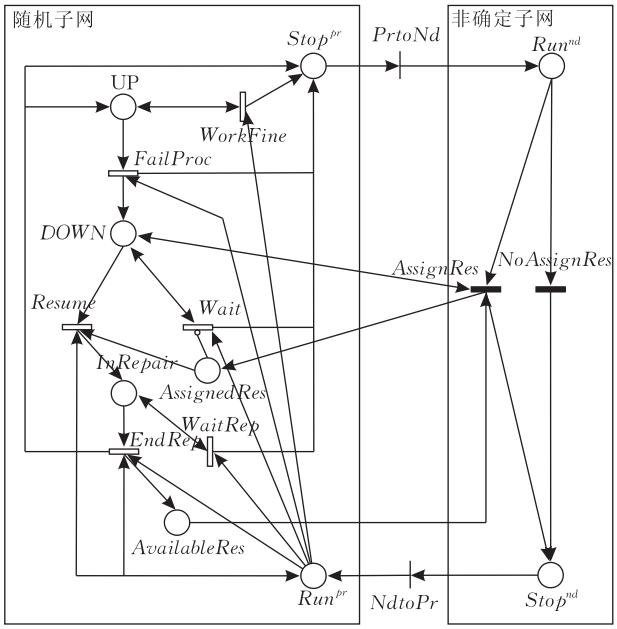


图 5 一个可修复系统的 MDWN 模型

不分配资源（变迁 *NoAssignRes*）. 在随机子网中，$Tstop^{pr}$ 类型的变迁有 *WorkFine*、*Fail*、*Wait*、*EndReq*，$Trun^{pr}$ 类型的变迁仅有一个，即 *Resume*. 非确定子网中所有的变迁均为 $Tstop^{nd}$ 类型.

在 MDPN 模型中，在位置、标记以及变迁中增加颜色的概念后，可进一步得到马尔可夫良构 Petri 网（Markov Decision Well-formed Nets，MDWN）模型. 这种模型可以较好地处理具有对称属性的系统，有效地缩小问题空间. MDPN 与 MDWN 的模型与算法都已经集成在 GreatSPN 工具中[43-45].

目前，针对 MDPN 与 MDWN 模型的应用研究已经逐步开展. 文献[46]分别利用 MDPN/MDWN 研究了高质量视频处理中的资源管理问题. 文献[47]研究了无线传感器网络中，对象移动跟踪的最优能源管理问题. 文献[48-49]研究了一类非确定可维修故障树（Non deterministic Repairable Fault Trees，NdRFT）模型与 MDWN 模型转换的方法，并将 MDWN 模型作为求解 NdRFT 模型最优策略的方法.

3.5　MDPN 与 ADP 的结合

3.3 小节中的 MDPN 求解方法，将 MDPN 规约为 MDP，然后再利用精确求解算法进行求解. 这种方式使得 MDPN 的求解仍然存在“状态空间爆炸”问题. 为此，我们将 MDPN 与 ADP 结合，利用 ADP 中 Monte Carlo 仿真的方法，解决 MDPN 的近似求解问题.

在结合 ADP 方法的 MDPN 中，不需要通过 Petri 模型得到完全的可达图 RG，也不需要通过式(42)计算状态之间的转移概率. 相反的，在模拟系统与决策者行为的同时，不断地更新可达状态集 RS. 由于 MDP 中只关注决策与系统运行的最终状态，因此只需记录位置 $Stop_i^{nd}$ 中全部都有标记的状态（称为决策终结状态），或者位置 $Stop_i^{pr}$ 中全部都有标记时的状态（称为系统终结状态）. 若该状态在 RS 中不存在，才将该状态加入 RS 中. 若新加入 RS 中的状态为决策终结状态，则为其关联一个后决策值函数并设定其初始值，其功能与式(25)类似.

进行仿真时，在每个系统终结状态可利用式(26)进行决策，并得到一个值函数样本. 注意行为 a_t 可能是一个行为序列. 此时，可利用式(35)，在值函数样本的基础上，更新其上一时刻决策终结状态的值函数. 这样，就将 ADP 中的前向动态规划算法集成到了 MDWN 中.

4　基于随机博弈网的动态优化模型

上述 MDP、MDPN 以及 MDWN 模型，都只能描述具有集中式控制设施的系统，即系统内只有一个决策者. 在现实生活中，还存在着大量具有多个决策者的系统. 上述模型在处理这类问题时，只能从各个决策者的角度分别建模，而将其他决策者视为不可控外部随机事件，无法体现出决策者之间的联系. 文献[50]于 2008 年首次提出了随机博弈网（Stochastic Game Nets，SGN），将动态随机博弈与随机 Petri 网结合，能够对具有多个决策者的系统进行建模分析.

动态随机博弈可以看作是马尔可夫决策过程的扩展，可包含多个决策者并能体现出他们之间的复杂关系，包括：(1) 竞争关系. 即每个决策者只关心最大化自己的收益；(2) 合作关系. 即所有决策者作为一个群体关心的是总收益. 将动态随机博弈与随机 Petri 网相结合，有助于系统的细粒度建模与简化求解.

4.1　随机博弈网

定义 3（随机博弈网）[51].　一个随机博弈网是一个 9 元组：$SGN=\{N,P,T,F,\pi,\lambda,R,U,M_0\}$，其中：

$N=\{1,2,\cdots,n\}$ 是决策者（博弈局中人）的集合；

P 是有限的位置集合；

$T=T^1\cup T^2\cup\cdots\cup T^n$ 是有限变迁的集合，其中 T^k 是第 $k\in N$ 个决策者的行为；

$\pi:T\to[0,1]$ 是决策者选择某个变迁的概率；

$F\subseteq I\cup O$ 是弧的集合，其中 $I\subseteq(P\times T)$，$O\subseteq(T\times P)$，且有 $P\cap T=\varnothing$，$P\cup T\neq\varnothing$. 记 x 的前集合为 ${}^{\cdot}x=\{y|(y,x)\in F\}$，$x$ 的后集合为 $x^{\cdot}=\{y|(x,y)\in F\}$；

$R:T\to(R_1,R_2,\cdots,R_N)$ 为决策者采用某个变迁所对应行为所得的收益函数，其中 $R_i\in(-\infty,+\infty)$，$i\in N$；

$\lambda=\{\lambda_1,\lambda_2,\cdots,\lambda_W\}$ 为变迁的实施速率，其中 W 是变迁的个数；

U 是决策者的总收益函数；

M_0 是起始状态，代表所有决策者的最初状态.

在该定义中，P 是博弈的状态，在某个位置 $p\in P$ 中有标记意味着所有决策者都在该状态中. 位置 p 中的标记 s 对应一个收益向量

$$\boldsymbol{h}_p(s)=(h_p^1(s),h_p^2(s),\cdots,h_p^k(s)) \tag{43}$$

其中 $h_p^k(s)$ 为决策者 k 在状态 p 中所得的收益. 当变迁 t 实施,所有决策者都会得到收益

$$\boldsymbol{R}(t)=(R_1(t),R_2(t),\cdots,R_k(t)) \tag{44}$$

其中 $R_i(t)$ 为决策者 i 所得的收益. 若标记经过变迁 t 到达位置 p,则收益都会累加在标记的收益向量 $\boldsymbol{h}_p(s)$ 中.

在 SGN 中,当系统运行至状态 p 时,决策者 k 的策略可定义为

$$\pi_k(p)=\{\pi(t_j^k)\}_{(t_j^k\in p^\cdot)\wedge(t_j^k\in T^k)} \tag{45}$$

其中,$\pi(t_j^k)$ 是决策者 k 采取变迁(即行为)t_j 的概率. 显然,对于所有状态 p,都有

$$\sum_{(t_j^k\in p^\cdot)\wedge(t_j^k\in T^k)}\pi(t_j^k)=1 \tag{46}$$

进一步,参照博弈论中纳什均衡的概念,可以定义 SGN 中的均衡策略 $\boldsymbol{\pi}^*=(\pi_1^*,\pi_2^*,\cdots,\pi_n^*)$ 满足

$$U_k(\pi_1^*,\cdots,\pi_{k-1}^*,\pi_k^*,\pi_{k+1}^*,\cdots,\pi_n^*)\geq U_k(\pi_1^*,\cdots,\pi_{k-1}^*,\pi_k,\pi_{k+1}^*,\cdots,\pi_n^*),\ \forall k\in[1,2,\cdots,n] \tag{47}$$

其中 π_k 是除 π_k^* 外所有其它可能的策略. 均衡策略的含义在于,某个决策者在其他决策者都不偏离均衡策略的情况下,采用非均衡策略不会取得比采用均衡策略更高的收益. 换句话说,该决策者没有偏离均衡决策的动机.

值得注意的是,在 MDP 中一般都使用确定行为(称为纯策略)作为最优解(一些例外的情况如探索/利用问题中会采取一些不确定行为来主动学习值函数). 而在 SGN 中,一般采用在行为空间的概率分布(混合策略)作为均衡解,因为在多人决策问题中,在纯策略意义下一般不存在均衡解,而在混合策略意义下一定存在均衡解.

4.2 随机博弈网的建模与分析

构建一个 SGN 模型一般分为 4 个步骤[52]:

(1) 建立每个决策者的子 SGN 模型.

在实际系统中,识别出 SGN 对应的要素,包括

① 变迁. 变迁代表决策者的行为. 注意行为集合中也可能包括空行为 ϕ,即决策者不采取任何行为.

② 收益. 对于每个变迁 t,赋予其一个收益函数 R,其每个分量 R_i 代表决策者 i 在该行为结束后所得的收益.

③ 位置集合 P. 每个位置 p 代表系统的一个状态.

(2) 描述纳什均衡条件.

对于竞争博弈,每个决策者的目标是最大化自己的收益;对于合作博弈,每个决策者的目标是最大化所有决策者的收益的总和. 对于有限时间 SGN,可仿照 MDP 中式(3)定义决策者 i 的总收益:

$$U_i^\pi=\mathrm{E}\Big\{\sum_{n=0}^{N}\alpha^n R_n^\pi\Big\} \tag{48}$$

其中 N 是时间的长度,R_n^π 是阶段 n 使用策略 π 时所得的收益. 注意 U_i^π 与所有决策者的策略都相关,而并非只与 i 自己的策略相关. 对于只有两个决策者的系统,均衡策略 $\pi^*=\{\pi^{1*},\pi^{2*}\}$ 满足 $U_1^{\pi^{1*},\pi^{2*}}\geq U_1^{\pi^1,\pi^{2*}}$ 且 $U_2^{\pi^{1*},\pi^{2*}}\geq U_2^{\pi^{1*},\pi^2}$.

(3) 求解纳什均衡策略.

一般情形下求解均衡策略难度较大. 本文仅对只有两个决策者的特殊情况进行讨论,此时,系统求解问题可以化归为一个静态非线性规划问题,详细请参见 4.3 小节.

(4) 合并子模型,建立全局 SGN 模型.

将子模型中含义相同的位置合并,可将所有子模型进行组合,得到全局 SGN 模型.

4.3 随机博弈网的求解

文献[52-54]给出了二人动态博弈的纳什均衡求解方法,该方法基于文献[55],将二人动态博弈问题化归为一个静态非线性规划(Non Linear Programming,NLP)问题:

$$\min_{U_1,U_2,\pi_1,\pi_2}\ \mathbf{1}^{\mathrm{T}}[U_k-R_k(\pi_1,\pi_2)-\alpha\boldsymbol{P}(\pi_1,\pi_2)U_k]$$

s.t.:

$$R_1(p_i)\pi_2(p_i)+\alpha\boldsymbol{T}(p_i,U_1)\pi_2(p_i)\leq\mathbf{1}^{\mathrm{T}}U_1(p_i),$$

$$(\pi_1(p_i))^{\mathrm{T}}R_2(p_i)+\alpha(\pi_1(p_i))^{\mathrm{T}}\boldsymbol{T}(p_i,U_2)\leq\mathbf{1}^{\mathrm{T}}U_2(p_i),$$

其中,值向量为

$$\boldsymbol{T}(p,U)=\{[\boldsymbol{P}(p_1|p,t^1,t^2),\cdots,\boldsymbol{P}(p_{|P|}|p,t^1,t^2)]^{\mathrm{T}}U_k\}$$

且有 $k\in\{1,2\}$,$i\in\{1,\cdots,|P|\}$,$p_i\in\boldsymbol{P}$,$t^1\in\boldsymbol{T}^1$,$t^2\in\boldsymbol{T}^2$. 该非线性规划的最小全局解,就是 SGN 中的纳什均衡解.

4.4 随机博弈网的模型化简与合并

当利用 SGN 对实际问题建模时,通常会遇到的一个问题是决策者的行为复杂,导致所建立的 SGN 模型难于求解分析. 文献[56]针对这一问题,给出了一些 SGN 模型化简的方法,例如在图 6 左半部分所示的模型,可以等价地化简为右半部分的简单模型.

4.2 小节中提到,在构建 SGN 全局模型时,需要进行子模型合并. 文献[57]讨论了在利用 SGN 对网络攻防进行建模时,子模型合并的方法,将决策者之间的关系分为两类:禁止类型与结束类型,相应的组合方法如图 7 所示.

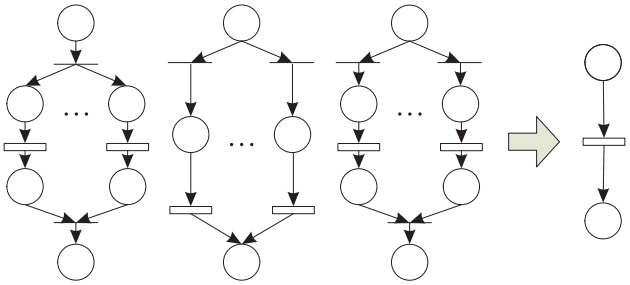
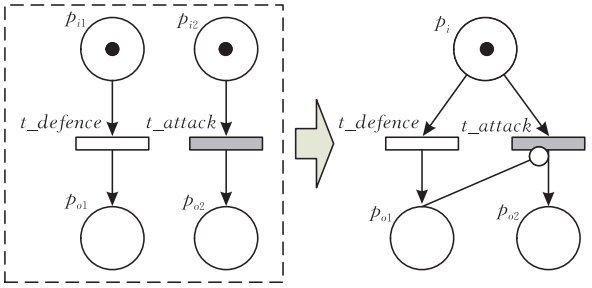
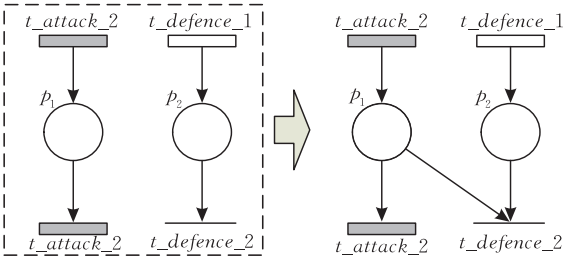


图 6　SGN 模型的化简



(a) 禁止类型



(b) 结束类型

图 7　模型的合并

一方面，当防御与攻击行为都可以实施时，若防御先实施，则可禁止攻击行为的实施(图 7(a))．另一方面，防御行为的实施，也可以使得整个攻击过程结束(图 7(b))．

4.5　应用与扩展

文献[57]在 SGN 的基础上进行延伸，进一步提出了针对网络安全攻防的攻击-防御随机博弈网(Attack-Defense Stochastic Game Nets，ADSGN)，准确地刻画了网络攻击者与防御者之间的零和竞争博弈关系．本小节中以企业网中的安全攻防问题为例，说明 SGN 的建模方法．

在一个典型企业网中，从攻击者与网络管理员的观点来看，网拓扑结构可抽象为图 8．攻击者可进行一些攻击行为，如扫描网络脆弱性、攻击数据库、破译服务器密码等．网络管理员可以进行一些相应的防御措施，例如利用入侵检测系统进行扫描、阻止攻击者 IP 进入系统、移除嗅探器等．攻击者 SGN 子模型与防御者 SGN 子模型分别如图 9、图 10 所示．这两个子模型从不同决策者的角度刻画了决策者在每个决策时间可能采取的攻防行为．应用 4.4 小节中的模型化简与合并技术，可将图 9、图 10 中的 SGN 子模型合并为图 11 所示的 SGN 完整模型．

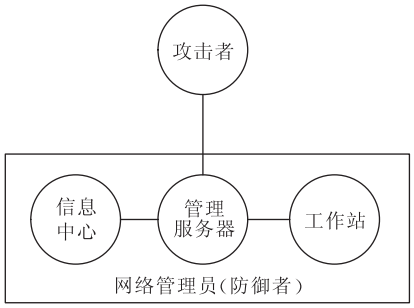


图 8　一个企业网网络拓扑结构

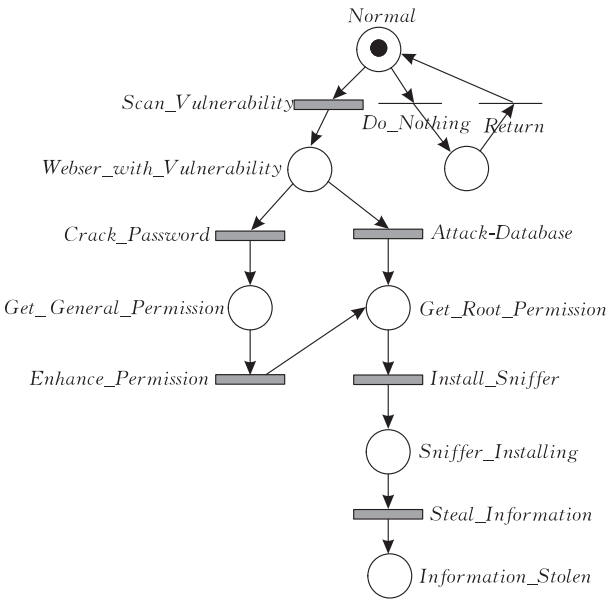


图 9　SGN 攻击者子模型

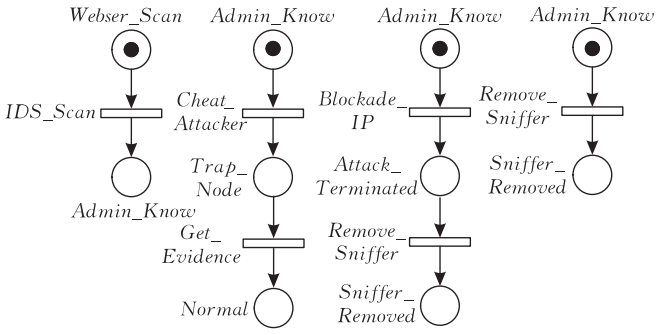


图 10　SGN 防御者子模型

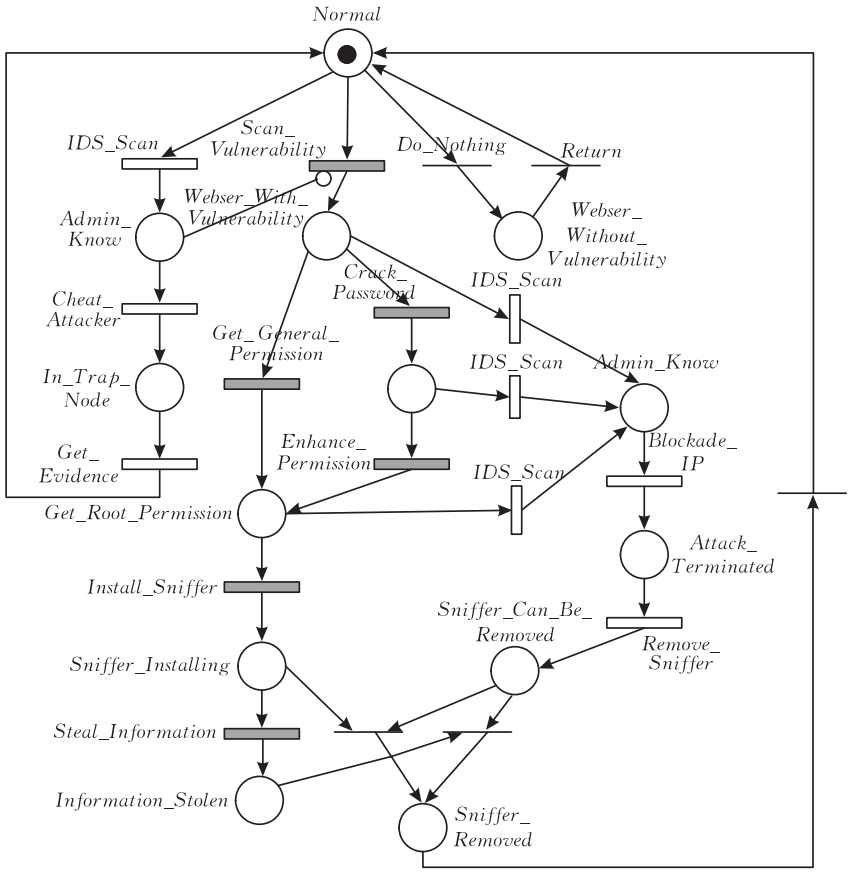


图 11 SGN 完整模型

目前,关于 SGN 的应用研究绝大多数都集中在网络安全方面,如文献[58]研究了利用网络连接关系与脆弱性信息等输入数据生成 SGN 模型的方法,文献[52-53,57]研究了企业网中的安全问题,文献[56,59]研究了电子商务中的若干安全问题,文献[54]研究了电子邮件蠕虫病毒的传播问题.另外,在无线网络领域,也有一些初步的研究成果,如文献[60]研究了无线网络中共享信道竞争的性能评价问题.总之,SGN 是一个正在发展与完善中的研究领域,在理论与应用方面均具有较为广阔的前景.

5 结论与展望

本文对动态优化在计算机系统与计算机网络中的建模、求解与应用进行了综述.相较于静态优化,动态优化可以精确地刻画系统的时变性.本文主要讨论了 3 种理论模型,即马尔可夫决策过程模型、马尔可夫决策 Petri 网模型以及随机博弈网模型,对这些模型的建模方法、求解算法、与应用实例进行了较为深入的研究.

计算机系统与计算机网络中的资源种类复杂,数量众多.面对这种复杂的应用环境,如何合理地运用动态优化理论对系统进行建模,并采取适当的求解算法进行(近似)求解具有极大挑战性.在本文最后以以下几点为例,列举一些未来可能的研究方向:

(1)马尔可夫决策过程的近似求解算法.众所周知,目前还不存在适用于所有 MDP 近似求解的统一"万能药"算法.很多看似合适的算法得出的近似解往往质量较差,在某些环境下甚至会出现算法不收敛的情况.近似解的质量在很大程度上还取决于算法设计者对领域专业知识的理解程度与算法设计经验,Powell 甚至将 ADP 近似值函数中的特征函数选取称为一种"艺术(art)"[36].对于近似求解算法的应用范围、解的质量以及收敛性等一系列问题,还需要进一步深入研究.

(2)马尔可夫决策 Petri 网与随机博弈网等模型的近似求解问题.一方面,在 MDPN/MDWN 与 SGN 模型中,虽然存在一些对模型进行化简的方法(如 4.4 节),但是这些方法往往局限于对某些特定模型结构的化简,还无法处理更为复杂的模型.另一方面,这些模型均采用精确求解算法,这使得利用这两种模型对大规模系统进行建模分析时求解较为困

难，大大限制了其应用范围. 在 3.5 节中我们对 MDWN 中结合 ADP 算法的方式进行了一些初步的探索，但还不够深入. 后续工作还应对这些模型的近似求解算法进行研究.

(3) 随机博弈网的应用研究拓展. 目前随机博弈网模型方法主要应用于网络安全分析中，而就随机博弈网的模型特点来说，它可以适用于模型分析具有多个独立决策者参与的计算机系统应用，如无线网络、对等网络(P2P)以及社交网络等. 进一步的研究工作将针对这些应用的特点，研究有针对性随机博弈网的建模与分析方法，拓展随机博弈网的应用领域.

参　考　文　献

[1] Murugesan S, Schniter P, Shroff N B. Multiuser scheduling in a Markov-modeled downlink using randomly delayed ARQ feedback. IEEE Transactions on Information Theory, 2012, 58(2): 1025-1042

[2] Zhao Qing, Swaml Anathram. A decision-theoretic framework for opportunistic spectrum access. IEEE Wireless Communications, 2007, 14(4): 14-20

[3] Zhao Q et al. Decentralized cognitive MAC for opportunistic spectrum access in Ad Hoc networks: A POMDP framework. IEEE Journal on Selected Areas in Communications, 2007, 25(3): 589-600

[4] Simunic Tajana, Benini Luca, Glynnt Peter, De Micheli Giovanni. Dynamic power management for portable systems// Proceedings of the 6th Annual International Conference on Mobile Computing and Networking. New York, USA, 2000: 11-19

[5] Srivastava Rahul, Koksal Can Emre. Energy optimal transmission scheduling in wireless sensor networks. IEEE Transactions on Wireless Communications, 2010, 9(5): 1550-1560

[6] Chen Huan, Huang Cheng-Wei. Power management modeling and optimal policy for IEEE 802.11 WLAN System// Proceedings of the IEEE 60th Vehicular Technology Conference. Los Angeles, USA, 2004: 4416-4421

[7] Choi Kae Won. Adaptive sensing technique to maximize spectrum utilization in cognitive radio. IEEE Transactions on Vehicular Technology, 2010, 59(2): 992-998

[8] Ouyang Wenzhuo, Murugesan Sugumar, Eryilmaz Atilla, Shroff Ness B. Exploiting channel memory for joint estimation and scheduling in downlink networks//Proceedings of the IEEE INFOCOM. Shanghai, China, 2011: 3056-3064

[9] Su Chi-Jiun, Tassiulas Leandros, Tsotras Vassilis J. Broadcast scheduling for information distribution//Proceedings of the IEEE INFOCOM'97. Sixteenth Annual Joint Conference of the IEEE Computer and Communications Societies. Kobe, 1997: 109-117

[10] Wang Rui, Lau V K N, Huang Huang. Delay optimal power control and relay selection for two-hop cooperative OFDM systems via distributive stochastic learning//Proceedings of the 2010 IEEE International Symposium on Information Theory (ISIT). Austin, Texas, 2010: 1843-1847

[11] Karmokar Ashok K, Djonin Dejan V, Bhargava Vijay K. Optimal and suboptimal packet scheduling over correlated time varying flat fading channels. IEEE Transactions on Wireless Communications, 2006, 5(2): 446-456

[12] Sutton R S, Barto A G. Reinforcement Learning: An Introduction. Cambridge, MA: The MIT Press, 1998

[13] Haas Zygmunt, Halpern Joseph Y, Li Erran L, Wicker Stephen B. A decision-theoretic approach to resource allocation in wireless multimedia networks//Proceedings of the 4th International Workshop on Discrete Algorithms and Methods for Mobile Computing and Communications. New York, USA, 2000: 86-95

[14] Rykov V V, Efrosinin D. Optimal control of queueing system with heterogeneous servers. Queueing Systems, 2004, 46(3-4): 389-407

[15] Maillart L M, Cassady C R, Rainwater C, Schneider K. Selective maintenance decision-making over extended planning horizons. IEEE Transactions on Reliability, 2009, 58(3): 462-469

[16] Chana G K, Asgarpoor S. Optimum maintenance policy with Markov processes. Electric Power Systems Research, 2006, 76(6-7): 452-456

[17] Djonin Dejan V, Krishnamurthy Vikram. MIMO transmission control in fading channels—A constrained Markov decision process formulation with monotone randomized policies. IEEE Transactions on Signal Processing, 2007, 55(10): 5069-5083

[18] Derman C, Klein M. Some remarks on finite horizon Markovian decision models. Operation Research, 1965, 13(2): 272-278

[19] Beutler F J, Ross K W. Optimal policies for controlled Markov chains with a constraint. Journal of Mathematical Analysis and Applications, 1985, 112(1): 236-252

[20] Thomas L C. Constrained Markov decision processes as multi-objective problems. Translated by White D J, French S, Hartley R. London: Academic Press, 1983

[21] Mihaylova L, Lefebvre T, Bruyninckx H, Gadeyne K, Schutter J D. A comparison of decision making criteria and optimization methods for active robotic sensing numerical methods and applications//Proceedings of the 1st European Symposium on Ambient Intelligence. Veldhoven, 2003: 316-324

[22] Chatterjee K. Markov decision processes with multiple long-run average objectives//Proceedings of the FSTTCS 2007. Lecture Notes in Computer Science 4855. Springer, 2007: 473-484

[23] Chatterjee K, Majumdar R, Henzinger T A. Markov decision processes with multiple objectives//Proceedings of the STACS 2006. Lecture Notes in Computer Science 3884. Springer, 2006: 325-336

[24] Karush W, Dear R E. Optimal strategy for item presentation in a learning process. Management Science, 1967, 13(11): 773-785

[25] Krishnamurthy V, Djonin D V. Structured threshold policies for dynamic sensor scheduling—A partially observed Markov decision process approach. IEEE Transactions on Signal Processing, 2007, 55(10): 4938-4957

[26] Chen Yunxia, Zhao Qing, Swami A. Joint design and separation principle for opportunistic spectrum access in the presence of sensing errors. IEEE Transactions on Information Theory, 2008, 54(5): 2053-2071

[27] Zhao Qing, Krishnamachari B, Liu Keqin. On myopic sensing for multi-channel opportunistic access: Structure, optimality, and performance. IEEE Transactions on Wireless Communications, 2008, 7(12): 5431-5440

[28] Liu Keqin, Zhao Qing, Krishnamachari B. Dynamic multichannel access with imperfect channel state detection. IEEE Transactions on Signal Processing, 2010, 58(5): 2795-2808

[29] Ahmad Sahand Haji Ali, Liu Mingyan, Javidi Tara, Zhao Qing, Krishnamachari Bhaskar. Optimality of myopic sensing in multichannel opportunistic access. IEEE Transactions on Information Theory, 2009, 55(9): 4040-4050

[30] Ji Shihao, Parr R, Carin L. Non-myopic multi-aspect sensing with partially observed Markov decision processes. IEEE Transactions on Signal Processing, 2005, 55(6): 2720-2730

[31] Kreucher C, Hero A, Kastella K. A comparison of task driven and information driven sensor management for target tracking//Proceedings of the 44th IEEE Conference on Decision and Control and 2005 European Control Conference (CDC-ECC'05). Spain, 2005: 4004-4009

[32] Liu Liming, Lu Yumao. Dynamic traffic controls for web-server networks. Computer Networks, 2004, 45(4): 523-536

[33] Givan R, Leach S, Dean T. Bounded parameter Markov decision processes. Artificial Intelligence, 2000, 122(1-2): 71-109

[34] Givan R, Leach S, Dean T. Bounded parameter Markov decision processes//Recent Advances in AI Planning. Lecture Notes in Computer Science 1348, 1997: 234-246

[35] Bertsekas D, Tsitsiklis J. Neuro-Dynamic Programming. Belmont, MA: Athena Scientific, 1996

[36] Powell W B. Approximate Dynamic Programming: Solving the Curses of Dimensionality. New York: John Wiley and Sons, 2007

[37] Roy B V, Bertsekas D P, Lee Y, Tsitsiklis J N. A neuro-dynamic programming approach to retailer inventory management//Proceedings of the 36th Conference on Decision and Control. San Diego, CA, 1997: 4052-4057

[38] Powell W B, George A, Ayari B B, Simao H P. Approximate dynamic programming for high dimensional resource allocation problems//Proceedings of the 2005 IEEE International Joint Conference on Neural Networks (IJCNN'05). Montreal, Canada, 2005, 5: 2989-2994

[39] George A, Powell W B, Kulkarni S R. Value function approximation using multiple aggregation for multiattribute resource management. Journal of Machine Learning Research, 2008, 9: 2079-2111

[40] Kaelbling L P, Littman M L, Moore A W. Reinforcement learning: A survey. Journal of Artificial Intelligence Research, 1996, 4: 237-285

[41] Beccuti M, Franceschinis G, Haddad S. Markov decision Petri net and Markov decision well-formed net formalisms. Technical Report, TR-INF-2007-02-01-UNIPMN, 2007, available via WWW at URL http: //www. di. unipmn. it/

[42] Beccuti M, Franceschinis G, Haddad S. Markov decision Petri net and Markov decision well-formed net formalisms//Proceedings of the 28th International Conference on Applications and Theory of Petri Nets and Other Models of Concurrency. 2007: 43-62

[43] Beccuti M, Raiteri D C, Franceschinis G, Haddad S. A framework to design and solve Markov decision well-formed net models//Proceedings of the 4th International Conference on Quantitative Evaluation of Systems. Washington DC, USA, 2007: 165-166

[44] Baarir S, Beccuti M, Cerotti D, Pierro M D, Donatelli S, Franceschinis G. The GreatSPN tool: Recent enhancements. ACM Performance Evaluation Review, 2009, 36(4): 4-9

[45] Beccuti M, Franceschinis G, Haddad S. MDWNsolver: A framework to design and solve Markov decision Petri nets. International Journal of Performability Engineering, 2011, 7(5): 417-428

[46] Beccuti M. Modeling and analysis of probabilistic systems: Formalisms and efficient algorithms [Ph. D. dissertation]. Dipartimento di Informatica, Universita degli Studi di Torino, 2009

[47] Beccuti M, Raiteri D C, Franceschinis G. Multiple abstraction levels in performance analysis of WSN monitoring systems//Proceedings of the 4th International ICST Conference on Performance Evaluation Methodologies and Tools. Brussels, Belgium, 2009: 1-10

[48] Beccuti M, Franceschinis G, Raiteri D C, Haddad S. Parametric NdRFT for the derivation of optimal repair strategies//Proceedings of the IEEE/IFIP International Conference on Dependable Systems & Networks. Lisbon, Portugal, 2009: 399-408

[49] Beccuti M, Raiteri D C, Franceschinis G, Haddad S. Non deterministic repairable fault trees for computing optimal repair strategy//Proceedings of the 3rd International Conference on Performance Evaluation Methodologies and Tools. Brussels, Belgium, 2008: 1-10

[50] Lin C, Wang Y Z, Wang Y. A stochastic game nets based approach for network security analysis//Proceedings of the 29th International Conference on Application and Theory of Petri Nets and Other Models of Concurrency, Concurrency Methods: Issues and Applications 2008 Workshop (Invited paper). Xi'an, China, 2008: 21-33

[51] Lin Chuang, Wang Yuan-Zhuo, Wang Yang. Stochastic Game Nets Based Network Security Evaluation and Analysis. Beijing: Tsinghua University Press, 2011(in Chinese)
(林闯，王元卓，汪洋. 基于随机博弈模型的网络安全评价与分析. 北京：清华大学出版社，2011)

[52] Wang Y Z, Lin C, Wang Y, Meng K. Security analysis of enterprise network based on stochastic game nets model//Proceedings of the 2009 IEEE International Conference on Communications (ICC'09). Dresden, Germany, 2009: 1-5

[53] Wang Y Z, Yu M, Li J Y, Meng K, Lin C, Cheng X Q. Stochastic game net and applications in security analysis for enterprise network. International Journal of Information Security, 2012, 11(1): 41-52

[54] Yu M, Wang Y Z, Liu Li, Cheng X Q. Modeling and analysis of email worm propagation based on stochastic game nets//Proceedings of the 12th International Conference on Parallel and Distributed Computing, Applications and Technologies (PDCAT'11). Gwanju, Korea, 2011: 381-386

[55] Filar J, Vrieze K. Competitive Markov Decision Processes. New York: Springer-Verlag, 1996

[56] Wang Y Z, Lin C, Meng K. Security analysis for online banking system using hierarchical stochastic game nets model//Proceedings of the GLOBECOM'09. Hilton Hawaiian Village, Honolulu, Hawaii, USA, 2009: 1-6

[57] Wang Y Z, Li J Y, Meng K, Lin C, Cheng X Q. Modeling and security analysis of enterprise network using attack-defense stochastic game net. Security Communication Networks, 2012

[58] Wang Yuan-Zhuo, Lin Chuang, Cheng Xue-Qi, Fang Bin-Xing. Analysis for network attach-defense based on stochastic game model. Chinese Journal of Computers, 2010, 33(9): 1748-1762(in Chinese)
(王元卓，林闯，程学旗，方滨兴. 基于随机博弈模型的网络攻防量化分析方法. 计算机学报，2010，33(9)：1748-1762)

[59] Wang Y Z, Lin C, Meng K, Lv J J. Analysis of attack actions for E-commerce based on stochastic game nets model. Journal of Computers, 2009, 4(6): 461-467

[60] Wan J X, Lin C, Chen X, Meng K, Wang Y Z. Performance analysis of channel contention in wireless Ad Hoc networks: A stochastic game nets approach//Proceedings of the GLOBECOM 2010. Miami, USA, 2010: 1-5

LIN Chuang, born in 1948, Ph. D., professor, Ph. D. supervisor. His research interests include computer networks, performance evaluation, network security analysis, and Petri net theory and its applications.

WAN Jian-Xiong, born in 1982, Ph. D. candidate. His research interests include performance evaluation and optimal control.

XIANG Xu-Dong, born in 1986, Ph. D. candidate. His research interests include performance evaluation and optimal control.

MENG Kun, born in 1980, Ph. D. candidate. His research interests include performance evaluation and stochastic models.

WANG Yuan-Zhuo, born in 1978, Ph. D, associate professor. His research interests include Petri net theory, and network security.

Background

Dynamic optimization theory is a powerful theoretical tool for modeling and solving sequential decision problems. It receives much attention from many academic areas such as operation research community, artificial intelligence community, and control theory community. This paper presents a brief overview of the models, solution techniques, as well as application of dynamic optimization theory in the field of computer systems and computer networks. Specifically, we discuss two kinds of extended dynamic optimization model, i. e., Markov Decision Petri Nets and Stochastic Game Nets.

This work is partly supported by the National Basic Research Program (973 Program) of China (Nos. 2010CB328105, 2009CB320505), National Natural Science Foundation of China (Nos. 60932003, 61070182, 60973144, 60973107, 61173008, 61070021). These projects aim to provide design principles for computer systems and computer networks from theoretical perspectives to improve the system performance and reduce the maintenance cost. Our group has been working on the performance evaluation and the optimization of the computer networks and computer systems for years. Many good papers have been published in respectable international conferences and transactions, such as INFOCOM, IEEE Journal on Selected Areas in Communication, IEEE Transactions on Information Theory, and IEEE Transactions on Signal Processing, etc. This paper summarizes these results and purposes some future research challenges.