面向医疗急救的信息网络服务功能链调度方法

张庆华[1],张先超[2,3],王寅昊[4],陆 军[5]

(1. 北京交通大学电子信息工程学院,北京 100044;2. 嘉兴学院浙江省医学电子与数字健康重点实验室,浙江嘉兴 314001;
3. 生命健康智能感知浙江省工程研究中心,浙江嘉兴 314001;4. 北京邮电大学信息与通信工程学院,北京 100876;
5. 中国电子科学研究院,北京 100041)

摘 要: 针对医疗急救分类分级服务需求,本文研究健康医疗信息网络服务功能链调度问题. 首先,将急救业务按照紧急程度和服务区域分为四类,设置优先级与权重,设计健康医疗信息网络架构、服务模式以及网络工作过程. 其次,以最小化总加权完成时间作为调度目标,建立服务功能链调度模型,针对不同问题规模,分别设计匹配博弈算法和*Q*-learning强化学习算法,求解功能链调度方案. 最后,开展仿真实验. 实验结果表明,本文的研究可以有效实现对医疗急救分级服务,能够推进健康医疗信息网络的建设,促进智慧医疗发展.

关键词: 健康医疗信息网络;医疗急救;服务功能链调度;总加权完成时间;匹配博弈;强化学习

基金项目: 国家自然科学基金(No.61941104);国家重点研发计划(No.2022YFC33014)

中图分类号: TP393 **文献标识码:** A **文章编号:** 0372-2112(2023)11-3128-09

电子学报 URL:http://www.ejournal.org.cn **DOI:**10.12263/DZXB.20230293

Service Function Chains Scheduling of Information Networks Used for Emergency Treatment

ZHANG Qing-hua[1], ZHANG Xian-chao[2,3], WANG Yin-hao[4], LU Jun[5]

(1. *School of Electronic Information Engineering, Beijing Jiaotong University, Beijing* 100044, *China*;
2. *Key Laboratory of Medical Electronics and Digital Health of Zhejiang Province, Jiaxing University, Jiaxing, Zhejiang* 314001, *China*;
3. *Engineering Research Center of Intelligent Human Situation Awareness of Zhejiang Province, Jiaxing, Zhejiang* 314001, *China*;
4. *School of Information and Communication Engineering, Beijing University of Posts and Telecommunications, Beijing* 100876, *China*;
5. *China Academic of Electronics and Information Technology, Beijing* 100041, *China*)

Abstract: To address the diverse emergency treatment needs, we investigate the scheduling problem of service function chains in healthcare networks. Firstly, emergency treatment services are categorized into four groups based on service area and urgency level, with each group assigned a corresponding priority and weight. We then propose a network framework, service model, and running process for healthcare networks. Secondly, we establish a service function chain scheduling model with the objective of minimizing the weighted total completion time. To address this problem at different scales, we develop a matching game algorithm and a *Q* reinforcement learning algorithm, both of which can yield an optimal scheduling scheme. Lastly, we conduct simulation experiments to assess our approach's effectiveness, which confirms that our research can enable classified and graded emergency treatment services, and facilitate the development of intelligent medical and healthcare networks.

Key words: healthcare networks; emergency treatment; service function chains scheduling; weighted total completion time; matching game; reinforcement learning

Foundation Item(s): National Natural Science Foundation of China (No.61941104); National Key Research and Development Program of China (No.2022YFC33014)

1 引言

近年来,健康信息化服务体系建设受到国家高度重视. 推进医疗卫生数据信息交换共享,动态调整服务资源配置,实现医疗数据共享和医疗分级服务是未来

收稿日期:2023-04-04;修回日期:2023-07-12;责任编辑:覃怀银

健康医疗信息网络的重点建设目标.

健康医疗信息网络,以物联网为基础,实时感知各类生命体征数据[1];利用互联网,实现生命体征数据在各级医疗单位间共享;利用云计算和大数据技术,为庞大生命数据提供强大的数据存储和分析处理能力[2]. 通过融合物联网、互联网、云计算与大数据技术,整合感知、通信、计算等资源,健康医疗信息网络可以跨越空间限制,按照医疗紧急程度动态调度全网资源[3]. 在健康医疗信息网络支持下,远程急救得到了应用和发展. 迫切需要建立可以依据患者危急程度和医疗区域进行分类分级急救的健康医疗信息网络,使得对于同等危急程度的患者,在远程达到与本地同等的救治率.

远程急救需要各类安全服务功能保护信息传输安全. 传统服务功能依靠各类专用硬件设备,服务能力有限,当业务数目超出设备能力范围,将导致服务效率严重下降[4]. 由于各类智能攻击手段不断升级,防火墙,入侵检测等安全设备难以为业务提供有效保护. 并且,硬件服务设备扩展、升级难度大,成本高昂[5]. 显然,传统服务模式不适用于健康医疗信息网络. 新兴网络功能虚拟化(Network Functions Virtualization, NFV)将网络功能以软件形式,布置在数据中心的服务器上,易于扩展、升级. NFV 还提供了一种灵活服务方式,业务所需的各类服务功能被编排成服务功能链(Service Function Chain, SFC),分配到服务器上处理[6]. 将 NFV+SFC 服务模式引入医疗信息网络可以满足其安全需求. 但是,在医疗信息网络中,医疗业务数量庞大,用户范围广,需要利用服务功能链调度技术,合理分配数据中心有限服务资源,提升服务效率[7].

服务功能链调度目前正处于起步阶段,是NFV领域的热点、难点[8]. 文献[9,10]首次在光网络中提出服务功能链调度问题,并给出了完整调度模型,优化目标是最小化最大完成时间,但并未给出具体求解方法. 以最小化最大完成时间作为优化目标的服务功能链调度问题多数是NP(Non-deterministic Polynomia)难问题,已有文献采用禁忌搜索[11]、遗传[12]等启发式算法求解服务功能链调度模型. 近年来,强化学习被越来越多地用于求解大规模优化问题,优势明显,文献[13]利用强化学习方法求解服务功能链调度问题. 以最小化最大完成时间作为优化目标的服务功能链调度问题未考虑单个服务功能链的完成时间,而在实际应用中说,不同服务功能链完成时间往往存在差异要求. 为了解决该问题,文献[14]在调度模型中为每个业务增加截止时间约束,既能降低服务链最大完成时间,又避免业务延期失效. 已有文献通过博弈[15]、匹配博弈[16]、强化学习[17]算法求解该问题.

综上,现有方法主要以最小化最大完成时间作为调度目标,确保最后完成的业务尽早完成. 但是,在医疗急救场景下,考虑到救治危急程度和距离远近,网络服务的优先级存在差异,优先级越高,越要确保其尽早完成. 以最小化最大完成时间为调度目标无法满足要求. 与已有工作相比,本文针对医疗急救需求,根据服务区域和紧急程度,对各类医疗业务分类分级,优先级越高,设置权重越大,以最小化总加权完成时间为调度目标建立服务功能链调度模型,设计匹配博弈算法和 Q-learning强化学习算法求解该问题,并开展模拟仿真实验.

2　适用分类分级急救的健康医疗信息网络

2.1　分类分级急救

根据紧急程度与服务区域,将医疗业务划分为四类,如表1所示.

表1　医疗业务及分类

服务区域	紧急程度	
	普通业务	紧急业务
本地业务	本地普通医疗业务	本地紧急医疗业务
远程业务	远程普通医疗业务	远程紧急医疗业务

(1)本地普通医疗业务:此类业务部署在医院内部,服务对象为非紧急患者,业务时效性要求相对较低. 主要包扩:普通患者健康数据收集/上传、辅助诊断等.

(2)远程普通医疗业务:此类业务需要医院间协作,传输距离远,需要保障时效性,但仍属于普通医疗业务,紧急性相对较低. 主要包括:远程问诊、远程科研协作、普通患者远程监护等.

(3)本地紧急医疗业务:此类业务部署在医院内部,服务对象为紧急患者,业务的时效性,紧急性远高于普通医疗业务. 主要包括:远程问诊、普通患者远程监护等.

(4)远程紧急医疗业务:此类业务需多方协作救治紧急患者,具有最高的时效性,紧急性,且传输距离远,需要给予最优先保障. 主要包括远程移动急救,院间协作急救等.

对四类业务划分优先级. 通常,紧急医疗优先级高于普通医疗,远程医疗优先级高于本地医疗,因此,业务的优先级和权重如表2所示.

2.2　健康医疗信息网络

健康医疗信息网络中四类医疗业务,需要网络提供

表2　医疗业务优先级

优先级	业务类型	权重
最高	远程紧急医疗业务	4
次高	本地紧急医疗业务	3
中等	远程普通医疗业务	2
普通	本地普通医疗业务	1

差异化、定制化的网络服务保障. 传统网络固化单一的服务模式在服务质量、资源管控等诸多方面存在严重弊端，难以支撑起分类、分级救治需求. 本文建立一种新型健康医疗信息网络架构，集成软件定义网络(Software Defined Networking，SDN)集中式管理，NFV+SFC服务模式，能按照业务的服务需求，动态灵活地调度网络资源[18].

如图1所示，健康医疗信息网络主要由数据中心、核心网、各级医院组成. 数据中心部署多台网络功能服务器，构成服务器集群，可以提供各类网络服务功能. 核心网连接数据中心与医院，主要由控制器和交换机组成，控制器统一管理、分配各类网络资源. 其中，服务功能链调度由集成在SDN控制模块中的服务功能调度子模块执行. 下级医院与中心医院部署有医疗业务所需的各类设备，包括医疗设备、通信设备等. 系统采用意图驱动设计思想，业务将以意图形式输入系统驱动网络运行[19]. 健康医疗信息网络还可以为急救车和居家医疗用户提供远程医疗服务.

健康医疗信息网络工作过程为：

(1)各级医院、救护车、居家医疗用户以意图的形式向控制器发送业务请求. 控制器通过北向接口获取请求，通过意图解析模块分析需求，通过服务功能调度模块生成服务功能调度策略，通过转发控制模块生成流量转发策略；

(2)根据服务功能调度策略和流量转发策略，控制器通过策略配置模块，生成网络配置文件，通过南向接口下发至数据平面；

(3)数据平面按配置文件转发业务流量至数据中心，数据中心为各类业务分配所需服务器资源；

(4)当业务的所有服务功能处理完成后，流量将被最终转发至目的节点，交付业务.

在健康医疗信息网络的工作过程中，数据中心处理各类业务所需的服务功能链，是保障整个网络高效运行的重要环节. 但是，由于数据中心的服务资源限制，以及急救业务的分类分级服务要求，需要研究服务功能链调度技术，为不同医疗业务提供差异化的最优调度方案.

3 服务功能链调度模型

网络中共有P个业务，任意业务$p(p\in\{1,2,\cdots,P\})$的优先级权重为ω_p，$\omega_p\in\{1,2,3,4\}$. P个业务的服务功能链集合为$S=\{s_1,s_2,s_3,\cdots,s_p,\cdots,s_P\}$. 第$p$个业务的服务功能链为$s_p$，$s_p=\{s_{p,1},s_{p,2},\cdots,s_{p,i},\cdots,s_{p,|s_p|}\}$，$i\in\{1,2,\cdots,|s_p|\}$，所有服务功能必须按顺序依次处理，即$s_{p,1}\prec s_{p,2}\prec,\cdots,\prec s_{p,i}\prec,\cdots,\prec s_{p,|s_p|}$

网络中共有M个服务器，这些服务器构成网络功能服务器集合$V=\{v_1,v_2,v_3,\cdots,v_m,\cdots,v_M\}$，

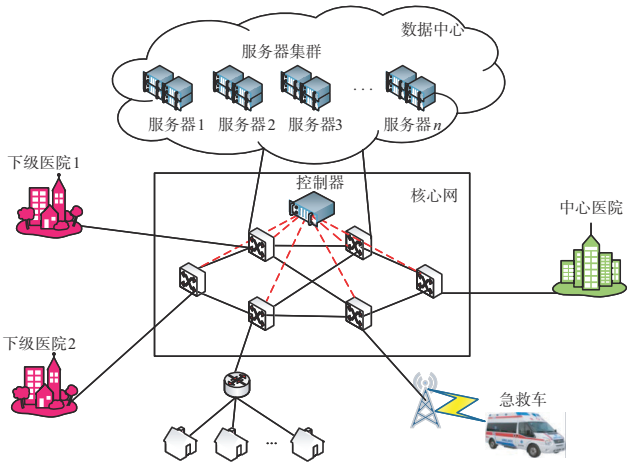


图1 健康医疗信息网络

$m\in\{1,2,\cdots,M\}$. 服务功能可以在每一台服务器上处理. 定义0，1变量$x_{m,p,i}$，$x_{m,p,i}=1$表示$s_{p,i}$分配给服务器v_m，否则$x_{m,p,i}=0$. 任意服务功能$s_{p,i}$只能分配给一台服务器处理，处理时间记为$t^{pr}_{p,i}$，开始处理时间为$t^{b}_{p,i}$，完成时间为$c_{p,i}$，服务链s_p的完成时间为C_p，则：

$$C_{p,i}=t^{b}_{p,i}+\sum_{m=1}^{M}x_{m,p,i}\cdot t^{pr}_{p,i} \tag{1}$$

$$C_p=t^{b}_{p,|s_p|}+\sum_{m=1}^{M}x_{m,p,|s_p|}\cdot t^{pr}_{p,|s_p|} \tag{2}$$

在相同服务功能链完成时间下，优先级越高，付出的代价也越大，应使高优先级业务尽早完成. 总加权完成时间可以代表业务的代价. 那么，可以用服务功能链总加权完成时间$T_P=\sum_{p=1}^{P}\omega_p\cdot C_p$作为医疗业务的代价. 建立服务功能链调度模型，

$$\min T_P=\sum_{p=1}^{P}\omega_p\cdot C_p$$

$$\text{s.t.}\begin{cases}
\text{C1: } t^{b}_{p,i}+t^{pr}_{p,i}\leqslant t^{b}_{p,i+1}\\
\text{C2: } t^{b}_{p,i}+z^{m}_{pi,qj}\cdot\sum_{m=1}^{M}x_{m,p,i}\cdot t^{pr}_{p,i}\leqslant t^{b}_{q,j}+z^{m}_{qj,pi}\cdot K\\
\text{C3: } t^{b}_{q,j}+z^{m}_{qj,pi}\cdot\sum_{m=1}^{M}x_{m,q,j}\cdot t^{pr}_{q,j}\leqslant t^{b}_{p,i}+z^{m}_{pi,qj}\cdot K\\
\text{C4: } z^{m}_{pi,qj}+z^{m}_{qj,pi}=1\\
\text{C5: } \sum_{m=1}^{M}x_{m,p,i}=1\\
\text{C6: } t^{b}_{p,i}\geqslant 0\\
\text{C7: } t^{b}_{p,j}\geqslant 0\\
\text{C8: } t^{b}_{q,j}\geqslant 0\\
\text{C9: } t^{pr}_{p,i}>0\\
\text{C10: } t^{pr}_{q,j}>0
\end{cases} \tag{3}$$

$$\forall p \in \{1,2,\cdots,P\}, \forall q \in \{1,2,\cdots,P\}, p \neq q,$$
$$\forall i \in \left\{1,2,\cdots,\left|s_p\right|\right\}, \forall j \in \left\{1,2,\cdots,\left|s_p\right|\right\},$$
$$\forall m \in \{1,2,\cdots,M\}, x_{m,p,i} = \{0,1\},$$
$$z^m_{pi,qj} = \{0,1\}, z^m_{qj,pi} = \{0,1\}$$

约束条件C1表示,所有服务功能必须按服务功能链的顺序执行,即前一项服务功能开始处理时间必须小于后一项服务功能开始处理时间.

约束条件C2~C4表示,如果两个服务功能链的服务功能被分配到了同一台服务器,那么,两个服务功能链的服务功能不能同时处理,必须在处理完成其中一个服务功能链的服务功能后,才可以处理另一服务功能链的服务功能. 因此,需要对服务功能进行排序,定义一个0,1辅助变量$z_{pi,qj}$:

$$z^m_{pi,qj} = \begin{cases} 1, & \text{如果} s_{p,i} \text{与} s_{q,j} \text{分配到同一台服务器,} \\ & \text{且} s_{p,i} \text{在} s_{q,j} \text{之前处理} \\ 0, & \text{其他} \end{cases} \tag{4}$$

其中,K为一个大正数,对于约束C2,如果$z_{pi,qj}=0$,$z_{pi,qj}=1$,则该约束失去限制性,对于约束C3如果$z_{pi,qj}=1$,$z_{pi,qj}=0$,则该约束失去限制性.

约束条件C5表示一个服务功能只能在一台服务器上处理.

约束条件C6~C8表示所有服务功能开始处理时间要大于或等于0.

约束条件C9、C10表示服务功能的处理时间要大于0.

求解最优服务功能链调度策略:

$\sigma^* = \left\{x^*_{m,p,i}, t^{b*}_{p,i}\right\}(\forall m \in \{1,2,\cdots,M\}, \forall p \in \{1,2,\cdots,P\},$
$\forall i \in \left\{1,2,\cdots,\left|s_p\right|\right\})$,使总加权完成时间最小.

4　服务功能链调度算法

不同问题规模对服务功能链调度算法求解质量和运行时间有着很大影响. 对于小规模问题,采用匹配博弈算法. 对于大规模问题,采用Q-learning强化学习算法.

4.1　匹配博弈服务功能链调度算法

服务功能链调度可以转化为服务功能与服务器一对一匹配博弈的结果. 将服务器和服务功能作为匹配博弈的双方,分别设计服务器匹配偏好列表和服务功能匹配偏好列表,按照业务优先级和服务器响应时间,对两个列表分别排序. 通过多轮博弈,获得满意的服务功能链调度策略.

将服务功能链调度转化为一个多轮匹配博弈过程,记总轮数为R,任意轮$r \in \{1,2,\cdots,r,\cdots,R\}$.

(1)匹配函数

匹配博弈对象为任意服务器v_m和服务功能,$s_{p,i}$对于任意轮r,服务功能链调度结果是一对一的匹配函数μ^r: $S \cup V \to 2^{S \cup V}$, 对于$\forall s_p \in S$, $\forall v_m \in V$, 当且仅当$\mu^r(s_{p,i}) = v_m$,并且$\mu^r(s_{p,i}) = v_m$时,服务功能$s_{p,i}$才会分配给服务器处理,即$x^r_{m,p,i}=1$.

(2)服务器匹配偏好列表

对于任意轮r,任意服务器m的匹配偏好列表为L^r_m. 每一轮列表中放入全部可以处理的功能. 服务器倾向于执行高优先级业务,所以,按照优先级权重由高到低对功能排序,服务器将按照偏好列表中的顺序选择当前轮服务功能.

(3)服务功能匹配偏好列表

对于任意轮r,任意服务功能$s_{p,i}$匹配偏好列表表示为$L^r_{p,i}$. 服务功能更倾向于选择响应时间小的服务器. 定义任意服务器m的服务器响应时间为o^r_m,

$$o^r_m = \begin{cases} 0, & \text{如果} r = 0 \\ o^{r-1}_m + \sum\limits_{p=1}^{P} \sum\limits_{i=1}^{|s_p|} x^{r-1}_{m,p,i} \cdot t^{pr}_{p,i}, & \text{其他} \end{cases} \tag{5}$$

按照服务器当前轮的服务响应时间由小到大的顺序,建立服务功能偏好列表.

(4)多轮匹配

最后,执行匹配操作,按照服务器匹配偏好列表和服务功能匹配偏好列表,将服务器与服务功能链的服务功能进行一对一匹配. 经过R轮匹配,所有服务功能与服务器完成匹配,生成优化调度方案.

流程如算法1所示.

4.2　Q-learning强化学习服务功能链调度算法

匹配博弈可以快速获得小规模问题的近似最优解,但不适用于求解大规模问题. 为了提升大规模问题的求解质量,设计Q-learning强化学习算法. 将第3节中的服务功能链调度问题转化为马尔科夫决策过程(Markov Decision Process, MDP)问题,由五元组$(\boldsymbol{S}, \boldsymbol{A}, \boldsymbol{P}, R, \gamma)$构成,$\boldsymbol{S}$为状态,$\boldsymbol{A}$为动作,$\boldsymbol{P}$为状态转移规则,$R$为奖励函数,$\gamma$为折扣因子.

(1)状态$\boldsymbol{S}$

调度总时间T被划分为若干个时隙,任意时隙$t \in [1,T]$时刻的状态为$\boldsymbol{s}_t = (\boldsymbol{E}(t), \boldsymbol{F}(t))$,$\boldsymbol{E}(t)$为$t$时隙的服务器状态,$\boldsymbol{E}(t) = (\boldsymbol{e}_1(t), \boldsymbol{e}_2(t), \cdots, \boldsymbol{e}_m(t), \cdots, \boldsymbol{e}_M(t))$. $\boldsymbol{F}(t)$为t时隙服务功能状态,$\boldsymbol{F}(t) = (\boldsymbol{f}_1(t), \boldsymbol{f}_2(t), \cdots, \boldsymbol{f}_l(t), \cdots, \boldsymbol{f}_L(t))$,$L$为网络中要处理的服务功能总数,$\boldsymbol{f}_l(t)$是$t$时隙,第$l$个要处理功能的状态.

$$\boldsymbol{e}_m(t) = \begin{cases} 0, & \text{如果服务器} v_m \text{没有处理功能} \\ 1, & \text{如果服务器} v_m \text{正在处理功能} \end{cases} \tag{6}$$

算法1 匹配博弈服务功能链调度算法

输入:服务链集合S,服务器集合V

输出:最小化加权完工时间T_P,最优服务功能链调度策略σ^*

1.初始化:$r\leftarrow 1,\{t^b_{p,i}=0\},\{x^r_{m,p,i}=0\},\{o^r_m=0\},\{c_{p,i}=0\},$

$(\forall m\in\{1,2,\cdots,M\},\forall p\in\{1,2,\cdots,P\},\forall i\in\{1,2,\cdots,|s_p|\})$

2.生成:$L^r_m,L^r_{p,i}$

3.$L^r_m\leftarrow s_{p,1}(\forall p\in\{1,2,\cdots,P\})$

4.对L^r_m中的服务功能按照优先级权重由高到低排序

5.将L^r_m中的服务功能依次分配给每一台服务器

6.从S中清除已分配的服务功能$s_{p',1}(\forall p'\in\{1,2,\cdots,P\})$

7.更新$x^r_{m',p',1}\leftarrow 1,c_{p',1}\leftarrow t^{pr}_{p',1},o^r_{m'}\leftarrow c_{p',1}$

8.第一轮匹配博弈完成,$r\leftarrow 2$

9.清空L^r_m中的所有服务功能

10.While存在未分配的服务功能

11. $L^r_m\leftarrow$每一服务功能链中排序第一的功能

12. 对L^r_m中的服务功能按照优先级权重由高到低排序

13. $s_{\text{best}}\leftarrow L^r_m$中排序第一的服务功能$s_{p',i'}$

14. $L^r_{p,i}\leftarrow$所有服务器

15. 对$L^r_{p,i}$中的服务器按照响应时间o^r_m由低到高排序

16. $v_{\text{best}}\leftarrow L^r_{p,i}$中排序第一的服务功能$v'_m$

17. If $i'\geqslant 2$, then

18. If ,$o^r_{m'}\geqslant C_{p',i'-1}$,then

19. $x^r_{m',p',i'}\leftarrow 1$

20. Else

21. $o^r_{m'}\leftarrow C_{p',i'-1}$

22. $x^r_{m',p',i'}\leftarrow 1$

23. Else

24. $x^r_{m',p',i'}\leftarrow 1$

25. 从S中清除已分配的服务功能$s_{p',i'}$

26. 更新$t^b_{p',i'}\leftarrow o^r_{m'},c_{p',i'}\leftarrow t^b_{p',i'}+t^{pr}_{p',i},o^r_{m'}\leftarrow c_{p',i'}$

27. 第r轮匹配博弈完成,$r\leftarrow r+1$

28. 清空L^r_m中的所有服务功能

$$\boldsymbol{f}_l(t)=\begin{cases}0, & \textit{如果服务功能}s_l\textit{未被处理}\\ 1, & \textit{如果服务功能}s_l\textit{正在处理}\\ 2, & \textit{如果服务功能}s_l\textit{已经处理}\end{cases}\tag{7}$$

(2)动作$\boldsymbol{A}$

智能体在状态$\boldsymbol{s}_t$下进行的动作为$\boldsymbol{a}_t$,$\boldsymbol{a}_t=(\boldsymbol{a}_1(t),\boldsymbol{a}_2(t),\cdots,\boldsymbol{a}_M(t))$,$\boldsymbol{a}_m(t)$表示$\boldsymbol{s}_t$状态下,服务器$v_m$进行的动作.$\boldsymbol{s}_t$状态下的动作为$\boldsymbol{A}(t)$,$\boldsymbol{A}(t)=\boldsymbol{A}_1(t)\otimes\boldsymbol{A}_2(t)\otimes,\cdots,\otimes\boldsymbol{A}_M(t)$. 其中,$\boldsymbol{A}_m(t)$为服务器$\boldsymbol{s}_t$状态下,服务器$v_m$所有可以进行的动作,$\otimes$为笛卡尔积.

为了找到不同状态$\boldsymbol{s}_t$下可执行动作,设计可行动作搜索算法,算法具体流程为:首先,建立空闲服务器,对于非空闲服务器,可行性动作为空. 然后,为每一个空闲服务器遍历网络中的所有服务功能,将$\boldsymbol{s}_t$下的所有可执行操作添加到$\boldsymbol{A}_m(t)$上. 最后,通过所有服务器的可执行动作的笛卡尔积,得到$\boldsymbol{s}_t$下的动作为$\boldsymbol{A}(t)$.

(3)状态转移规则$\boldsymbol{P}$

网络每经过一个时隙时间,更新一次系统状态. 网络初始状态为$\boldsymbol{s}_1$,$\boldsymbol{s}_1=\boldsymbol{s}_{\text{ini}}$,网络的结束状态为$\boldsymbol{s}_{\text{end}}$. 对于任意状态$\mathbf{s}_t$,$\boldsymbol{E}(t)$的状态迁移为

$$\boldsymbol{e}_m(t+1)=\begin{cases}0, & \textit{如果}\boldsymbol{a}_m(t)=0,\boldsymbol{e}_m(t)=1,\boldsymbol{\theta}_m(t)=1\\ 1, & \textit{如果}\boldsymbol{a}_m(t)\neq 0,\boldsymbol{e}_m(t)=1,\boldsymbol{\theta}_m(t)>1\\ \boldsymbol{e}_m(t), & \text{其他}\end{cases}\tag{8}$$

其中,$\boldsymbol{\theta}_m(t)$表示在t时隙服务功能在服务器v_m上剩余的处理时间.$\boldsymbol{e}_m(t+1)=0$表示,在$t+1$时隙,服务器完成上一功能处理.$\boldsymbol{e}_m(t+1)=1$表示在$t+1$时隙,服务器仍在处理上一功能.

$\boldsymbol{F}(t)$的状态迁移表示为

$$\boldsymbol{f}_l(t+1)=\begin{cases}1, & \textit{如果}\boldsymbol{f}_l(t)=0,l\in\boldsymbol{a}_t,\boldsymbol{\tau}_l(t)>1\\ 2, & \textit{如果}\boldsymbol{f}_l(t)=0,l\in\boldsymbol{a}_t,\boldsymbol{\tau}_l(t)=1\\ 2, & \textit{如果}\boldsymbol{f}_l(t)=1,\boldsymbol{\tau}_l(t)>1\\ \boldsymbol{f}_l(t), & \text{其他}\end{cases}\tag{9}$$

其中,$\boldsymbol{\tau}_l(t)$表示,在t时隙,$\boldsymbol{f}_l$剩余处理时间.$\boldsymbol{f}_l(t+1)=1$表示$\boldsymbol{f}_l$在$t+1$时隙将被处理,且处理时间大于1.$\boldsymbol{f}_l(t+1)=2$分两种情况,第一种,$\boldsymbol{f}_l$将在$t+1$时隙处理,且处理时间为1;第二种,$\boldsymbol{f}_l$在t时隙正在处理,且剩余处理时间大于1.

(4)奖励函数R

设置累积奖励为总加权完成时间的倒数,

$$R(s_t,a_t)=1/\sum_{p=1}^{P}\omega_p\cdot C_p,\quad\forall(s_t,a_t)\in\Omega\tag{10}$$

(5)算法流程

使累积奖励最大化的最优调度策略为σ^*:

$$\sigma^*=\max_{\sigma}\sum_{t\in[1,T]}R(\boldsymbol{s}_t,\boldsymbol{a}_t)\tag{11}$$

使智能体通过与系统不断交互来学习最优调度策略. 维护调度策略表(Q-table),在每一轮迭代结束时,更新Q-table.

$$\begin{aligned}Q(\boldsymbol{s}_t,\boldsymbol{a}_t)=&(1-\alpha)Q(\boldsymbol{s}_t,\boldsymbol{a}_t)+\alpha(R(\boldsymbol{s}_t,\boldsymbol{a}_t)\\&+\gamma\max_{\boldsymbol{a}_{t+1}}Q(\boldsymbol{s}_{t+1},\boldsymbol{a}_{t+1}))\end{aligned}\tag{12}$$

其中,α是学习率.α和γ是取值为0~1之间的实数.$Q(\boldsymbol{s}_t,\boldsymbol{a}_t)$是$t$时隙$(\boldsymbol{s}_t,\boldsymbol{a}_t)$的期望奖励. 假设$Q$表收敛到最优$Q^*$,则可以通过贪婪探索得到最优策略$\sigma^*$,即

$$\sigma^*=\arg\max_{\boldsymbol{a}_t}Q^*(\boldsymbol{s}_t,\boldsymbol{a}_t)\tag{13}$$

当网络中所有服务器都在处理服务功能时,不需要执行动作,只在执行动作的时隙更新Q-table. 执行动作时隙间隔为Δt. 根据执行动作的时隙,将式(12)改为

$$Q(\boldsymbol{s}_t,\boldsymbol{a}_t)=(1-\alpha)Q(\boldsymbol{s}_t,\boldsymbol{a}_t)+\alpha(R(\boldsymbol{s}_t,\boldsymbol{a}_t)+\gamma\max_{\boldsymbol{a}_{t+\Delta t}}Q(\boldsymbol{s}_{t+\Delta t},\boldsymbol{a}_{t+\Delta t})) \tag{14}$$

流程如算法2所示.

算法2　Q-learning强化学习服务功能链调度算法

输入:服务链集合S,服务器集合V

输出:最小化加权完工时间T_p,最优服务功能链调度策略σ^*

1.初始化:时间t=0,episode数n=0,最大episode数$n_{\max}$,系统初始状态$\boldsymbol{s}_{\text{ini}}$,探索率$k$,学习率$\alpha$,折扣因子$\gamma$

2.根据服务链集合S计算系统最终状态$\boldsymbol{s}_{\text{end}}$

3.While $n<n_{\max}$

4.　重置时间$t\leftarrow 0$

5.　重置服务链完成时间C_p

6.　重置系统状态$\boldsymbol{s}_t\leftarrow\boldsymbol{s}_{\text{ini}}$,

7.　重置动作集$\Omega_n\leftarrow\varnothing$

8.　While $\boldsymbol{s}_t\neq\boldsymbol{s}_{\text{end}}$

9.　　$\boldsymbol{V}(t)\leftarrow$寻找当前时隙所有可用服务器

10.　　$\boldsymbol{A}(t)\leftarrow$寻找当前时隙所有可用动作

11.　　While True

12.　　if $\boldsymbol{V}(t)\neq\varnothing$ and $\boldsymbol{A}(t)\neq\varnothing$

13.　　以概率k为$\boldsymbol{V}(t)$中的节点随机选取一个可取动作,

14.　　否则,为节点选择Q表中对应概率最高的动作

15.　　将做过分配的节点和对应VNF从$\boldsymbol{V}(t)$和$\boldsymbol{A}(t)$中移除

16.　　else

17.　　break

18.　　将剩余节点动作设置为空动作并得到动作集$\boldsymbol{a}_t$

19.　　如果$\boldsymbol{a}_t$不是空动作,则将$\boldsymbol{s}_t$,$\boldsymbol{a}_t$添加至Ω_n

20.　　$\boldsymbol{s}_{t+1}\leftarrow$根据采取的动作更新系统状态

21.　　$t\leftarrow t+1$

22.　$n\leftarrow n+1$

23.　对所有Ω_n中的所有状态动作对计算奖励并更新Q表

24.　根据式(10)计算奖励

25.　根据式(14)更新Q表

26.$k\leftarrow 0$

27.$\sigma^*\leftarrow\Omega_{n_{\max}}+1$

5　仿真分析

5.1　仿真设置

仿真实验硬件设备为配备Intel Core(TM) i9-11900K@2.8 GHz CPU的计算机,仿真软件为python3.7. 设置6组算例,每个算例设置如表3所示. 每一业务包含1条服务功能链,每个服务链中的服务功能数$|s_p|\in\{2,3,4,5\}$,处理时间$t_{p,i}^{pr}\in\{1,2,3,4,5\}$. 6组算例中,设置四个业务等级,各等级业务数量如表4所示. Q-learning强化学习算法episode集数设置为2 000,探索率k和学习率α设置为0.1,折扣因子γ设置为0.8.

5.2　算法比较

本文在6组算例下,分别利用匹配博弈、强化学习、遗传、差分进化、模拟退火五种算法求解服务功能链调度模型,总加权完成时间性能对比结果如表5、表6.

表3　服务器和业务数量设置

参数	算例1	算例2	算例3	算例4	算例5	算例6
服务器/台	3	3	9	9	20	20
业务/个	40	400	40	400	40	400

表4　各类业务数量　　单位:个

参数	算例1	算例2	算例3	算例4	算例5	算例6
远程紧急业务	10	100	10	100	10	100
本地紧急业务	10	100	10	100	10	100
远程普通业务	10	100	10	100	10	100
本地普通业务	10	100	10	100	10	100

由表5,对于小规模问题(业务数量为40个),总加权完成时间为10^3数量级. 表中,H_M表示匹配博弈算法比其他算法在总加权完成时间上降低的百分比. 可以看出,匹配博弈算法的总加权完成时间比Q-learning强化学习、遗传、模拟退火三种算法至少降低了4.5%,至多降低了26.6%. 这是因为,匹配博弈算法通过设置服务功能和服务器偏好列表,在满足匹配规则前提下,从服务功能和服务器博弈双方角度决定调度方案,兼顾了二者需要满足的匹配条件,避免了调度决策的随机性,在解空间较小情况下,可以在短时间内得到高质量的调度方案. 但是,随着问题规模增加,解空间也随之增大,由于匹配博弈算法严格按照匹配规则决定调度方案,使算法无法寻找匹配规则之外的其他解,难以达到全局最优.

由表6,对于大规模问题(业务数量为400个),总加权完成时间为10^5数量级. 表中,H_Q表示Q-learning强化学习算法比其他算法在总加权完成时间上降低的百分比. Q-learning强化学习算法的总加权完成时间比匹配博弈、遗传、模拟退火三种算法至少降低了2.1%,至多降低了40.0%. 这是因为,Q-learning强化学习算法,通过不断自学习,逐渐改进智能体的动作决策,利用探索过程避免算法陷入局部最优. 此外,差分进化算法在遗传算法的基础上,通过缩放操作,确保变异后可以得到不同的个体,提升了探索能力,更容易实现全局最优,所以在6组算例下均有着最高求解质量. 但是,其算法复杂度和算法运行时间高于其他算法.

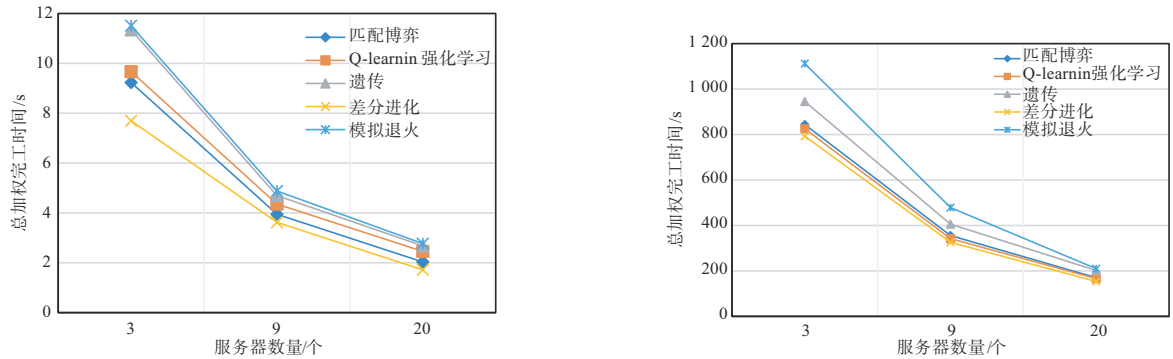
算法在不同服务器数量下的总加权完成时间对比如图2. 从图2可以更加直观看出,对于小规模问题,匹配博弈算法的总加权完成时间在所有服务器数量下,均低于Q-learning强化学习算法和遗传、模拟退火两种对比算法,仅高于差分进化算法. 对于大规模问题,Q-learning强化学习算法的总加权完成时间在所有服务器数量下,均低于匹配博弈算法和遗传、模拟退火两种对比算法,仅高于差分进化算法. 由图2还可以看出,服

表5 小规模问题总加权完成时间性能对比

算法	算例1		算例3		算例5	
	T_P/ms	H_M/%	T_P/ms	H_M/%	T_P/ms	H_M/%
匹配博弈	9 230	—	3 945	—	2 032	—
Q-learning强化学习	9 666	4.5	4 354	9.3	2 457	17.3
遗传	11 342	18.6	4 694	16.0	2 677	24.1
差分进化	7 695	−19.9	3 621	−8.9	1 716	−18.4
模拟退火	11 512	19.8	4 878	19.1	2 770	26.6

表6 大规模问题总加权完成时间性能对比

算法	算例2		算例4		算例6	
	T_P/ms	H_Q/%	T_P/ms	H_Q/%	T_P/ms	H_Q/%
匹配博弈	843 666	2.1	355 538	3.8	170 025	2.3
Q-learning强化学习	825 340	—	342 018	—	165 986	—
遗传	945 474	12.7	405 778	15.7	201 934	17.8
差分进化	792 836	−4.0	325 866	−4.9	153 559	−8.0
模拟退火	1 112 099	25.8	478 731	40.0	209 940	23.0



(*a*) 算法求解小规模问题总加权完工时间对比 (*b*) 算法求解大规模问题总加权完工时间对比

图2 算法求解不同规模问题总加权完工时间对比

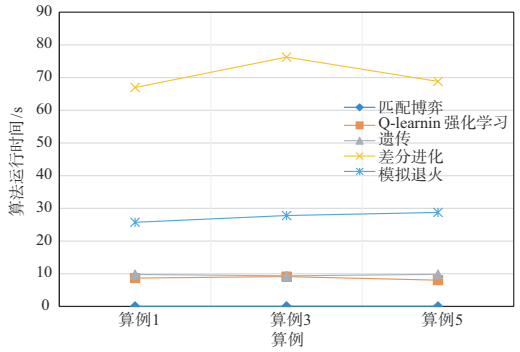
务器数量越少,算法在总加权完成时间性能上的差异越明显,这是因为在理想状态下,如果业务数量小于服务器数量,那么所有业务都可以同时处理. 资源越有限,服务功能链调度作用越大. 差分进化算法在两种问题规模中都有着最小的总加权完成时间,但算法的运行时间高于其他算法. 算法运行时间如表7所示.

表7 算法运行时间对比 单位:s

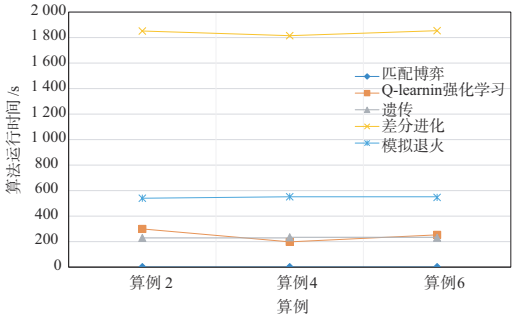
算法	算例1	算例2	算例3	算例4	算例5	算例6
匹配博弈	0.013	0.15	0.04	0.14	0.02	0.14
Q-learning强化学习	8.68	299.53	9.13	198.24	8.04	253.14
遗传	9.78	229.16	9.37	234.22	9.81	230.06
差分进化	66.98	1 851.43	76.26	1 814.91	68.84	1 854.01
模拟退火	25.74	540.33	27.81	551.72	28.76	546.69

由表7可知,在6组算例中,匹配博弈算法运行时间低于其他四种算法,且都可以在“毫秒级”时间完成,随着网络规模的增大,时间为多项式增长. 匹配博弈算法的复杂度为$O(L_s)$. 强化学习算法、遗传算法和模拟退火算法复杂度均为$O(L_s^2)$. 差分进化算法复杂度为$O(L_s^3)$,L_s为服务功能总数. 匹配博弈算法复杂度最

低,运行时间最短,差分进化算法复杂度最高,运行时间最长. 虽然强化学习算法与遗传算法和模拟退火算法有着相同的复杂度,在运算时间上的优势并不明显,但其求解质量高于这两种算法. 由上述结果,本文采用的匹配博弈算法和Q-learning强化学习算法,可以快速求解不同规模下的服务功能链分类分级调度问题. 不同规模下的算法运行时间对比如图3所示.



(*a*) 算法求解小规模问题算法运行时间对比



(*b*) 算法求解大规模问题算法运行时间对比

图3　算法求解不同规模问题运行时间对比

从图3可以更加直观地看出,在两种规模问题下,差分进化算法的运行时间最高,匹配博弈算法的运行时间最低.

6　结束语

本文研究了面向紧急救治的健康医疗信息网络服务功能链分类分级调度,将医疗业务进行分类、分级,建立健康信息网络架构,明确网络工作过程. 建立了以最小化总加权完成时间为目标的服务功能链调度模型,并设计了求解该问题的匹配博弈算法和Q-learning强化学习算法. 开展了仿真实验,对总加权完成时间,算法运行时间两方面进行比较. 本研究可用于医疗急救行业,能够打破地理空间限制,实现中心医院—社区医院—救护车等多方协同急救,促进医疗急救分类分级标准建立. 对于信息网络行业,本研究可以支持数据中心服务功能调度系统等核心设备研制,促进网络功能虚拟化技术投入实际运营使用,减少运营成本,提升服务弹性. 本文的研究内容可以促进医疗行业和信息网络行业融合发展.

参考文献

[1] CAO R H, TANG Z, LIU C B, et al. A scalable multicloud storage architecture for cloud-supported medical internet of things[J]. IEEE Internet of Things Journal, 2020, 7(3): 1641-1654.

[2] PAKDEL R, HERBERT J. Scalable cloud-based analysis framework for medical big-data[C]//2016 IEEE 40th Annual Computer Software and Applications Conference (COMPSAC). Piscataway: IEEE, 2016: 647-652.

[3] BHUIYAN M N, RAHMAN M M, BILLAH M M, et al. Internet of Things (IoT): A review of its enabling technologies in healthcare applications, standards protocols, security, and market opportunities[J]. IEEE Internet of Things Journal, 2021, 8(13): 10474-10498.

[4] KHÄN T, BAUMGARTNER A, BAUSCHERT T. Optimising virtual network functions migrations: A flexible multi-step approach[C]//2019 IEEE Conference on Network Softwarization (NetSoft). Piscataway: IEEE, 2019: 188-192.

[5] DONG L, FONSECA N L S DA, ZHU Z Q. Application-driven provisioning of service function chains over heterogeneous NFV platforms[J]. IEEE Transactions on Network and Service Management, 2021, 18(3): 3037-3048.

[6] LI J, LIANG W F, MA Y. Robust service provisioning with service function chain requirements in mobile edge computing[J]. IEEE Transactions on Network and Service Management, 2021, 18(2): 2138-2153.

[7] 兰巨龙, 金子晋, 孙鹏浩, 等. 基于可靠性的服务功能链构建算法[J]. 通信学报, 2019, 40(1): 64-70.
LAN J L, JIN Z J, SUN P H, et al. Service function chain construct algorithm based on reliability[J]. Journal on Communications, 2019, 40(1): 64-70. (in Chinese)

[8] 祖家琛, 胡谷雨, 严佳洁, 等. 网络功能虚拟化下服务功能链的资源管理研究综述[J]. 计算机研究与发展, 2021, 58(1): 137-152.
ZU J C, HU G Y, YAN J J, et al. Resource management of service function chain in NFV enabled network: A survey [J]. Journal of Computer Research and Development, 2021, 58(1): 137-152. (in Chinese)

[9] RIERA J F, ESCALONA E, BATALLÉ J, et al. Virtual network function scheduling: Concept and challenges[C]// 2014 International Conference on Smart Communications

in Network Technologies (SaCoNeT). Piscataway: IEEE, 2014: 1-5.
[10] RIERA J F, HESSELBACH X, ESCALONA E, et al. On the complex scheduling formulation of virtual network functions over optical networks[C]//2014 16th International Conference on Transparent Optical Networks (ICTON). Piscataway: IEEE, 2014: 1-5.
[11] MIJUMBI R, SERRAT J, GORRICHO J L, et al. Design and evaluation of algorithms for mapping and scheduling of virtual network functions[C]//Proceedings of the 2015 1st IEEE Conference on Network Softwarization (NetSoft). Piscataway: IEEE, 2015: 1-9.
[12] QU L, ASSI C, SHABAN K. Delay-aware scheduling and resource optimization with network function virtualization[J]. IEEE Transactions on Communications, 2016, 64(9): 3746-3758.
[13] WANG T F, ZU J C, HU G Y, et al. Adaptive service function chain scheduling in mobile edge computing via deep reinforcement learning[J]. IEEE Access, 2020, 8: 164922-164935.
[14] ALAMEDDINE H A, QU L, ASSI C. Scheduling service function chains for ultra-low latency network services [C]//2017 13th International Conference on Network and Service Management (CNSM). Piscataway: IEEE, 2018: 1-9.
[15] ALAMEDDINE H A, SEBBAH S, ASSI C. On the interplay between network function mapping and scheduling in VNF-based networks: A column generation approach [J]. IEEE Transactions on Network and Service Management, 2017, 14(4): 860-874.
[16] PHAM C, TRAN N H, HONG C S. Virtual network function scheduling: A matching game approach[J]. IEEE Communications Letters, 2018, 22(1): 69-72.
[17] LI J L, SHI W S, ZHANG N, et al. Delay-aware VNF scheduling: A reinforcement learning approach with variable action set[J]. IEEE Transactions on Cognitive Communications and Networking, 2021, 7(1): 304-318.
[18] REN C, LI H, LI Y X, et al. On efficient service function chaining in hybrid software defined networks[J]. IEEE Transactions on Network and Service Management, 2022, 19(2): 1614-1628.
[19] KIRAN M, POUYOUL E, MERCIAN A, et al. Enabling intent to configure scientific networks for high performance demands[J]. Future Generation Computer Systems, 2018, 79: 205-214.

作者简介

张庆华 男,1994年2月出生于吉林省长春市. 北京交通大学博士研究生. 主要研究方向为网络资源管理、服务功能链调度等.
E-mail: 20111019@bjtu.edu.cn

张先超(通讯作者) 男,1984年3月出生于安徽省合肥市. 嘉兴学院教授,东南大学博士后. 主要研究方向为网络资源管理、人工智能等. 中国电子学会会员编号:E190019119M.
E-mail: zhangxianchao@zjxu.edu.cn

王寅昊 男,1998年11月出生于山西省晋中市. 北京邮电大学硕士研究生. 主要研究方向为网络功能虚拟化、服务功能链调度等.
E-mail: 13363541648@163.com

陆 军 男,1964年11月出生于江苏省苏州市. 中国工程院院士. 主要研究方向为综合电子信息系统,人工智能. 中国电子学会会员编号:E190136424M.
E-mail: lujun_cetc@163.com