Journal of Computer Applications
计算机应用,2015,35(8):2375 -2379
ISSN 1001-9081
CODEN JYIIDU
2015-08-10
http://www.joca.cn

文章编号:1001-9081(2015)08-2375-05
doi:10.11772/j.issn.1001-9081.2015.08.2375

基于微重启和部分客观马尔可夫决策模型的智能水下机器人软件自修复方法

张汝波[1,2],孟　雷[1],史长亭[1*]
(1.哈尔滨工程大学 计算机科学与技术学院,哈尔滨 150001;　2.大连民族大学 机电工程学院,辽宁 大连 116600)
(*通信作者电子邮箱 shichangting@hrbeu.edu.cn)

摘　要:针对智能水下机器人(AUV)软件故障修复过程中存在的修复代价过高和系统环境只有部分可观察的问题,提出了一种基于微重启技术和部分客观马尔可夫决策(POMDP)模型的 AUV 软件故障修复方法。该方法结合 AUV 软件系统分层结构特点,构建了基于微重启的三层重启结构,便于细粒度的自修复微重启策略的实施;并依据部分可观马尔可夫决策过程理论,给出 AUV 软件自修复 POMDP 模型,同时采用基于点的值迭代(PBVI)算法求解生成修复策略,以最小化累积修复代价为目标,使系统在部分可观环境下能够以较低的修复代价执行修复动作。仿真实验结果表明,基于微重启技术和 POMDP 模型的 AUV 软件故障修复方法能够解决由软件老化及系统调用引起的 AUV 软件故障,同与两层微重启策略和三层微重启固定策略相比,该方法在累积故障修复时间和运行稳定性上明显更优。

关键词:智能水下机器人;微重启;自修复;部分客观马尔可夫决策;基于点的值迭代算法
中图分类号:TP311.52　　**文献标志码**:A

Self-repair method for autonomous underwater vehicle software based on micro-reboot and partially observable Markov decision process model

ZHANG Rubo[1,2], MENG Lei[1], SHI Changting[1*]
(1. *College of Computer Science and Technology, Harbin Engineering University, Harbin Heilongjiang* 150001, *China*;
2. *College of Mechanical and Electronic Engineering, Dalian Nationalities University, Dalian Liaoning* 116600, *China*)

Abstract: Aiming at the disadvantages of high fixing cost and partial observability of system environment in the process of repairing Autonomous Underwater Vehicle (AUV) software faults, a method was proposed based on micro-reboot mechanism and Partially Observable Markov Decision Process (POMDP) model for failure repair of AUV. To facilitate the implementation of the fine-grained self-repair micro-reboot strategy, a hierarchical structure was built based on micro-reboot combined with the characteristics of AUV software. Meanwhile, a self-repair model was put forward according to the theory of POMDP. With the goal of minimizing the fixing cost, the repair strategy was solved by Point Based Value Iteration (PBVI) algorithm to allow the repair action to execute in the partially observable environment at a lower cost. The simulation results show that the proposed repairing method can solve the AUV software failures caused by the software-aging and system calls. Compared with two-tier micro-repair strategy and three-tier micro-repair fixing strategy, this method is obviously superior to the contrast method in cumulative fault repair time and operational stability.

Key words: Autonomous Underwater Vehicle (AUV); micro-reboot; self-repair; Partially Observable Markov Decision Process (POMDP); Point Based Value Iteration (PBVI) algorithm

0　引言

随着智能水下机器人(Autonomous Underwater Vehicle, AUV)智能化和独立完成特定任务的能力逐步提高,其嵌入式实时控制软件的复杂性也随之不断增强,对 AUV 软件系统的可靠性提出了更高要求[1]。由于 AUV 运行在深海高危环境中,这种环境下如何准确高效修复突发性故障进而提高系统稳定性已成为重要研究课题。自修复技术是解决这类问题有效途径,国内外学者在这方面做了大量工作。目前软件自修复所针对系统按应用场景可以分为:安全性要求不高的系统和安全稳定性要求高的系统。第一类系统主要用在商业服务和关键业务上,但安全稳定性要求不高,比如 Web 服务器或数据库管理系统(Database Management System, DBMS)就是这类系统。文献[2]针对这类系统提供了一个明确包含维修、操作、质量属性的修复框架;文献[3]针对 Web 服务提出了一种多混合模型自恢复解决方法;文献[4]则针对自组织结构多代理系统(Multi-Agent System, MAS)提出了一种细观随机修复方法,这种方法被应用于分布式自修复资源系统中。第二类主要是对安全稳定性要求高的系统,比如运行 AUV 软件的 VxWorks 嵌入式系统。文献[5]设计了一套基于 VxWorks 系统的快速故障恢复方法用于修复通信系统基站故障;文献[6]设计并实现了一种在 VxWorks 下基于检测点技

收稿日期:2015-03-25;**修回日期**:2015-06-03。　**基金项目**:装备预研基金项目(9140C270101130C27099)。
作者简介:张汝波(1963 -),男,吉林吉林人,教授,博士生导师,博士,主要研究方向:人工智能、机器学习、计算智能、模式识别;　孟雷(1987 -),男,安徽宿州人,硕士研究生,主要研究方向:软件可靠性、机器学习;　史长亭(1980 -),男,山东茌平人,讲师,博士,主要研究方向:软件可靠性、智能控制。

术的软件修复方法;文献[7]对星载 VxWorks 系统相关软件可靠性及修复技术进行了研究;文献[8]设计了一套基于 VxWorks 的高可容错性软件系统。虽然软件自修复方面不断涌现出新研究成果,但往往都主要针对于第一类系统研究,对于第二类系统,特别是专门针对 VxWorks 这类实时操作系统软件自修复技术的相关研究较少。

传统自修复技术主要采用基于容错的方法,包括多版本编程技术、恢复块技术、检测点技术等,这些方法修复代价大并且无法满足用户对软件系统的高可靠性需求。针对传统修复方法缺陷,Candea 等[9]提出了采用递归重启方式解决软件运行故障的微重启技术。近年来微重启技术用于故障修复研究也取得了丰富成果:文献[10]中探讨了微重启方法,针对微重启需要克服系统底层软件带来的技术问题,给出了不同类别不同复杂性的解决方案;文献[11]中提出了一种针对树状结构跟踪算法的重启案例,针对于重启这种修复手段,将其具体细分为多粒度修复,并创新性定义了应用组件、应用进程、操作系统、虚拟机监控和物理节点五个细粒度级别;文献[12]中对目前现有的细粒度修复手段进行了评估,包括独立的应用程序重启、与热备份结合的应用程序重启、系统级重启、系统级快速重启、虚拟机重启和物理节点重启等;文献[13]将神经网络应用到重启技术中,降低了确定重启相关度和可达集的复杂性;文献[14]将微重启技术应用到细分策略的软件抗衰老修复中;文献[15]将微重启技术用于 AUV 软件系统中,提出了解决嵌入式平台系统软件故障的修复架构。由于微重启技术能很好地修复 AUV 软件的瞬时或间歇性故障失效,并且实现简单修复成本低,因此本文选用微重启技术解决 AUV 软件故障修复问题。同时,由于 AUV 软件内部真实故障状态是无法完全感知的,在这种故障状态部分可观情况下如何选取最佳修复动作序列属于部分可观马尔可夫决策过程(Partially Observable Markov Decision Process, POMDP)策略求解,因此本文选用 POMDP 模型解决 AUV 软件自修复策略学习问题。

1 基于三层重启结构的微重启方法

AUV 软件系统主要用于 AUV 运行期间对各种作业执行、任务规划和航线选择进行决策控制,AUV 软件系统是 AUV 最高执行中枢,是发出各种命令的决策者。AUV 作为一种高智能水下运载器,其软件结构具有层次性强、耦合度高等特点,以本文研究的 AUV 为例,其软件运行于 VxWorks 嵌入式系统平台上采用分层和模块化思想组织代码,整体结构分为任务层、模块层和系统层。因此,根据 AUV 软件结构特点和 VxWorks 系统平台中任务是其运行最小独立单位的特点,同时为了有效减小自修复代价,更好实现微重启修复效果,本文将 AUV 智能决策系统自修复重启层次划分为三个重启层:任务层、模块层和系统层。任务层修复操作指的是对本层内各种任务进行微重启;模块层修复操作指的是对本模块内所有任务进行统一微重启;系统层修复操作指的是对 AUV 软件进行整体重启,相当于宏重启操作,使 AUV 软件整体重新初始化运行。图 1 展示了 AUV 智能决策系统自修复微重启层次结构。

AUV 软件出现故障时依靠如图 1 所示的微重启层次结构进行从下往上递归重启。首先重启任务层中相关故障任务,若无法解决故障问题则重启上层模块层内所有任务,若仍然无法解决故障问题则再次往上层移动重启系统层,也就是进行宏重启。在这种递归式层次结构中,任务层、模块层、系统层修复代价依次递增,并且每层修复代价都相差很多。在不同重启层次上进行相应微重启模型构建,最终可建立整体 AUV 软件自修复微重启模型。

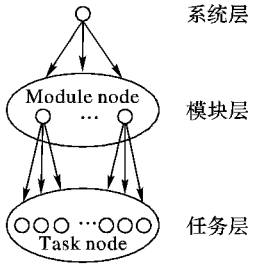

图 1 AUV 智能决策系统自修复微重启层次结构

2 基于 POMDP 的 AUV 软件自修复模型

俄罗斯数学家 Andrei Andreyevich Markov 首次提出马尔可夫决策模型,它是指一类具有马尔可夫性质的数学决策模型。若系统通过决策执行动作 a_t 转移到下一个状态 s_{t+1},仅依赖于当前状态 s_t 而与之前历史状态无关,这种动态系统决策过程称为马尔可夫决策过程(Markov Decision Process, MDP)。假设 Agent 在马尔可夫环境中执行动作决策,但只能通过观测值间接获得环境状态信息,这种有别于 MDP 的模型,称之为 POMDP 模型,可用七元组描述 $\langle S,A,Tr,R,Z,O,b_0\rangle$。其中:$S$ 代表环境状态集,A 代表代理可执行的一系列动作集,Tr 是状态转移函数,R 是奖赏函数,Z 观测值集,O 观测转移函数,b_0 是初始信度状态。POMDP 模型是匹配真实环境中 Agent 作出决策的最佳模型,而 AUV 软件自修复问题也可以理解成自修复代理根据 AUV 软件状态信息通过决策选取最佳修复动作执行,这样的自修复决策过程具有马尔可夫性,因此可使用 POMDP 模型对 AUV 软件自修复过程进行数学描述,具体构建如下。

1)状态空间。

通过分析 AUV 软件故障影响模式,将 AUV 软件自修复 POMDP 模型状态空间指定为失效发生的具体位置。具体状态空间以任务为单位定义为:

$$S = \{s_0, s_1, s_2, s_3, \cdots, s_n, s_{n+1}, \cdots, s_m, s_{m+1}, \cdots, s_k, s_p\} \quad (1)$$

其中:$s_i(i \in (1,n))$ 表示 AUV 软件系统中第 i 个任务发生故障;$s_j(j \in (n+1,m))$ 表示各模块故障;$s_q(q \in (m+1,k))$ 表示同时存在两个或两个以上任务失效状态;s_0 表示 AUV 软件系统中各任务均正常无故障发生,s_p 表示发生系统级故障。

2)动作空间。

结合基于三层的微重启修复技术,具体动作空间定义为:

$$A = \{a_1, a_2, a_3, \cdots, a_n, a_{n+1}, \cdots, a_m, a_s\} \quad (2)$$

其中各动作具体含义如表 1 所示。

表 1 修复执行动作表

代号	具体修复方法	类别
$a_{ij}(i \in [1,n], j \in [1,2])$	微重启或重启第 i 个任务	任务层
$a_i(i \in [n+1,m])$	重启第 $i-n$ 个模块	模块层
a_s	重启系统	系统层

3)观测空间。

观测值通过 AUV 软件自修复体系结构中故障检测模块

输出值确定,具体观测空间定义如下:

$$Z = \{z_0, z_1, z_2, \cdots, z_n\} \tag{3}$$

其中:除 z_0 表示 AUV 软件处于正常状态外,其余观测值分别表示 AUV 软件各模块发生故障。

4)概率分布。

AUV 软件整体自修复 POMDP 模型概率分布包含状态转移函数和观测值转移函数两类。

a)状态转移函数。由于在当前状态下执行指定动作后,下一时刻状态无法完全确定,这种状态转移不确定性可以使用状态转移函数描述,具体定义为:

$$Tr(s_j, s_i, a_i) = P(S(t+1) = j \mid S(t) = i, a_i \in A), \quad \forall s_i, s_j \in S \tag{4}$$

表示在状态 s_i 下执行动作 a_i 转移到下一时刻状态 s_j 的概率。

b)观测函数。观测值主要通过故障检测模块获得,同样故障检测模块也可能由于某些因素导致输出错误结果或观测不完全,因此需要对观测值作概率性定义:

$$O(z, a_i, s_j) = P(Z(t+1) = z \mid S(t+1) = s_j, a_i \in A), \quad \forall s_j, s_i \in S, \forall z \in Z \tag{5}$$

表示在 t 时间步执行动作 a_i 转移到状态 s_j 获得观测值 z 的概率。

5)修复代价。

在 AUV 软件自修复 POMDP 模型中,修复代价函数主要由修复动作执行时间 T_a、当前故障单元风险程度 K 和修复结果奖惩值 Q 共同决定。风险程度指在若干失效的后验概率相差不大且恢复时间相近时,优先选择能够恢复具有较高风险失效状态的动作,从而提高系统可靠性。修复结果奖惩值指当执行动作达到预期时给予系统奖励值,否则给予惩罚值,从而确保系统具有自学习能力。POMDP 模型就是在整体修复效果最佳情况下保证累计修复代价 R 最低,具体定义为:

$$R(B_n(s), a) = T_a * \ln(K+1) - Q \tag{6}$$

修复代价函数 R 的偏置条件是:假定每次修复动作 a 都能正常执行。在实际中若修复动作 a 未正常执行,可考虑设置重复执行阈值,超出阈值仍未正常执行则进行系统级修复动作,或按任务硬性故障处理。具体的修复代价确定方法可以通过在 VxWorks 系统平台上对各任务和模块进行故障注入,然后逐一对每个任务及模块执行微重启记录有效重启时间 T_a,同时结合具体的故障单元风险程度 K 和修复结果奖惩值 Q 代入式(6)中求得。

6)信度。

根据状态空间划分形式,定义 AUV 软件自修复模型信度状态如下:

$$B(s) = \{b(s_0), b(s_1), b(s_2), b(s_3), \cdots, b(s_n), b(s_p)\} \tag{7}$$

其中:$b(s_i)$ 表示在当前环境下 AUV 软件系统处于 s_i 状态信度,信度状态是定义在部分可观状态空间 S 上的概率分布。

3　基于 POMDP 的 AUV 软件自修复策略规划

基于 POMDP 的 AUV 软件自修复模型选取修复动作的依据是最小化代价函数 R。通过对 POMDP 模型求解 π^*,可得到使累计代价值最小的最优策略,即 AUV 软件自修复决策模型为:

$$V_{n+1}(b) = \min_{a \in A} \left\{ \sum_{s \in S} b(s) R(s,a) + \gamma \sum_{z \in Z} \sum_{s \in S} b(s) \sum_{s' \in S} b(s) Tr(s', s, a) O(z, a, s')) V_n(b') \right\} \tag{8}$$

其中 $V_\pi(b)$ 表示在信念状态 b 下策略 π 的值函数。式(8)表示在有限修复步骤内,积累代价值最小的动作为最优决策动作。

为确保自修复系统稳定性,选择离线规划求解模型策略。在离线 POMDP 求解算法选取上,考虑到 AUV 自修复问题特点和代码实现的复杂性,同时借鉴相关领域学者已有研究成果选取值迭代算法的近似解法——基于点的值迭代(Point Based Value Iteration, PBVI)算法。PBVI 算法伪码如下所示,其执行过程可简述为:1)使用 Init(*POMDP*)初始化信度状态集合 B,值函数集 V,其中 *POMDP* 指具体 POMDP 模型参数;2)对每个信度状态集合 B 中元素 b 使用 Backup (b, V) 进行回溯操作更新值函数;3)使用 Expand (B) 扩充信度状态集合。

PBVI 算法伪代码如下:

```
PBVI (POMDP, ITERATIONS)
{
  B, V ← Init (POMDP)
  While True do
    For n in ITERATIONS
      For b in B
        α ← Backup(b, V)
        Add (V, α)
      End for
    End for
    B ← Expand(B);
  End while
}
```

为了确保 PBVI 算法运行,需要事先求解 AUV 软件自修复 POMDP 模型的状态转移概率和观测值转移概率。本文主要依赖于专家经验和历史数据构建自修复模型状态转移概率、观测值转移概率。离线情况下 AUV 软件自修复问题历史数据结构为 (s_t, a_t, z_t, R_t) 四元组组成的历史序列。其中状态 s_t 可通过分析 AUV 软件代码等方式进行人工标注。当具有一定数量历史数据时,可利用极大似然估计方法估计 AUV 自修复模型参数,具体如下:

1)状态转移概率 Tr_{iaj} 估计。设样本中 t 时刻处于状态 i 执行动作 a 后 $t+1$ 时刻转移到状态 j 的频数为 A_{iaj},那么状态转移概率 Tr_{iaj} 的估计是:

$$Tr_{iaj} = A_{iaj} \Big/ \sum_{j=s_0}^{|S|} A_{iaj};\ i \in S, j \in S, a \in A \tag{9}$$

2)观测值概率 O_{iaz} 的估计。设样本中状态 i 执行动作 a 后观测为 z 的频数为 B_{iaz},那么观测值概率 O_{iaz} 的估计是:

$$O_{iaz} = B_{iaz} \Big/ \sum_{z=z_0}^{|Z|} B_{iaz};\ i \in S, a \in A, z \in Z \tag{10}$$

4　实验及结果分析

4.1　自修复效果评价实验

为了验证本文提出的基于三层微重启方案及使用 PBVI 算法求解出的修复策略实际执行效果,进行仿真实验。实验分两个步骤:首先比较基于 POMDP 的三层微重启自修复策

略、两层重启修复策略、最优修复策略三种修复方法实际修复效果;其次,在同样使用三层微重启修复方案中,比较基于 POMDP 的自修复方案与其他固定策略修复方案的实际修复效果。

4.1.1 三层微重启方案效果

实验方案:依据初始信度 b_0 通过对 AUV 软件故障注入方式构建故障样例 1 000 个,使用基于 POMDP 的三层微重启自修复策略、两层重启修复策略、最优修复策略分别对同样的 1 000 个故障测试样例进行故障修复,评价修复效果。其中:两层重启修复策略为只使用模块和系统进行重启的传统修复策略,最优修复策略假定系统可以获知当前真实故障状态进而可选取最佳修复动作。

图 2 和图 3 分别展示了针对于任务规划模块故障样例和紧急情况处理模块故障样例下三种修复策略累积修复代价。可以看出,基于 POMDP 的三层微重启策略累积修复代价均小于二层微重启策略,基于 POMDP 的三层微重启策略累积修复代价曲线更接近于最优策略。

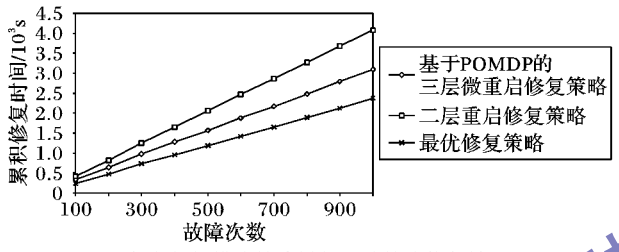

图 2 任务规划模块故障样例三种策略修复效果对比

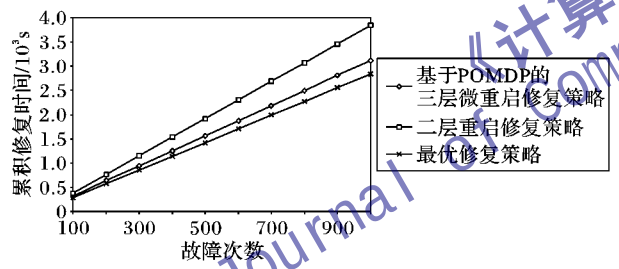

图 3 紧急情况处理模块故障样例三种策略修复效果对比

4.1.2 基于 POMDP 的 AUV 软件自修复策略效果

实验方案:依据初始信度 b_0 通过对 AUV 软件故障注入方式构建故障样例 1 000 个,使用基于 POMDP 的三层微重启自修复策略、基于三层微重启的固定策略(因动作序列组合不同,会有多种策略形式)、分别对同样的 1 000 个故障测试样例进行故障修复,记录每种方案修复 1 000 个故障样例的累积修复代价,然后比较每种修复方案的累计修复代价大小,评价修复效果。对任务规划模块和紧急情况处理模块构建两组故障测试样例,在两组测试样例中分别进行上述实验比较不同方案修复效果,结合两组实验数据综合评价确保实验结果准确。

图 4 和图 5 分别展示了针对于任务规划模块故障样例和紧急情况处理模块故障样例下三层微重启修复策略累积修复代价,修复完 1 000 个故障样例时累积修复代价如表 2 所示。其中三层微重启修复固定策略二和基于 POMDP 的三层微重启修复策略由于修复动作序列差异较小累积修复代价较接近,图 5 中累积修复曲线趋于重叠。通过该实验可知,基于 POMDP 的三层微重启策略累积修复代价均小于其他三层微重启固定策略。

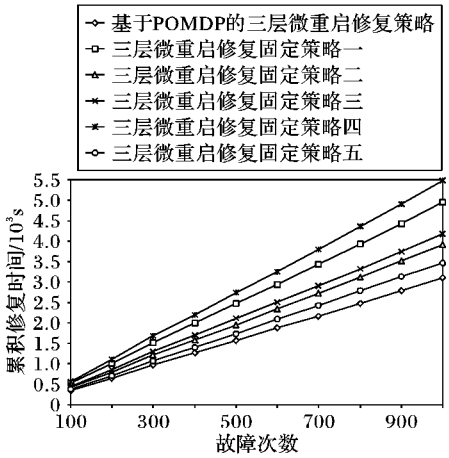

图 4 任务规划模块故障样例三层微重启策略修复效果对比

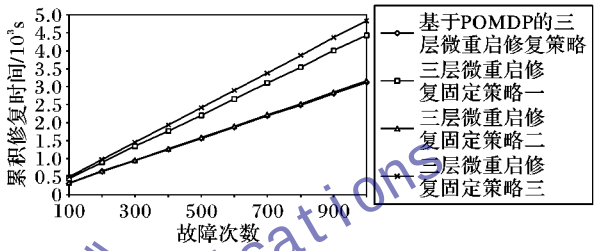

图 5 紧急情况处理模块故障样例三层微重启策略修复效果对比

表 2 1000 个故障样例累积修复时间

策略	任务规划模块故障修复时间/s	紧急情况处理模块故障修复时间/s
基于 POMDP 三层微重启修复策略	3 099.5	3 116
三层微重启修复固定策略一	4 958.5	4 431
三层微重启修复固定策略二	3 917.5	3 166
三层微重启修复固定策略三	4 186	4 839
三层微重启修复固定策略四	5 488	
三层微重启修复固定策略五	3 468	

4.2 自修复稳定性评价实验

实验方案:通过分析 4.1 节实验数据,以每 100 次实验为单位,求解累积修复代价均值,比较各种修复策略平均累积修复代价波动情况,以评价哪种修复策略修复效果稳定性更好。对任务规划模块和紧急情况处理模块构建两组故障测试样例分别进行上述实验,结合两组实验数据综合评价确保实验结果准确。

从图 6 和图 7 中可以看出,对每个故障样例而言,基于 POMDP 的三层微重启修复策略修复代价都高于最优修复策略而低于所有其他修复策略。考察两幅图中每种修复策略整体曲线变化趋势,在图 6 中相对于最优修复策略来说,三层微重启修复固定略一、三层微重启修复固定策略二、三层微重启修复固定策略三、基于 POMDP 的三层微重启修复策略曲线变化幅度均很大,这是由于这几种策略动作序列较长,各动作修复代价差别大引起的,在图 7 中可以发现同样现象。两幅图中在 300 次故障修复之前各修复策略累积修复代价均值都有较大起伏变化,这主要是由于故障样例构建时,在 300 次故障样例中没有完全符合初始信度中各故障概率分布,这样可能存在某个故障样例区间内多次出现系统层或模块层故障,导致修复代价均值出现跳跃。但是在这种不规则分布故障样例中,本文设计的方案均能在花费相对较小修复代价情况下

修复故障问题。因此可以得出结论,在故障状态发生概率基本符合初始信度时,基于 POMDP 的三层微重启修复策略相对稳定性较好,平均修复代价相对其他策略最低。

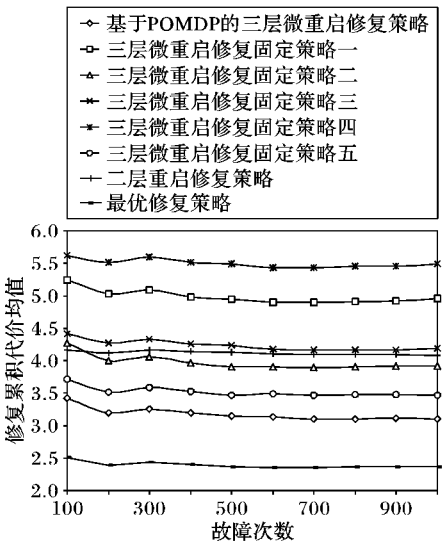

图 6　任务规划模块故障样例各策略修复累积代价均值变化

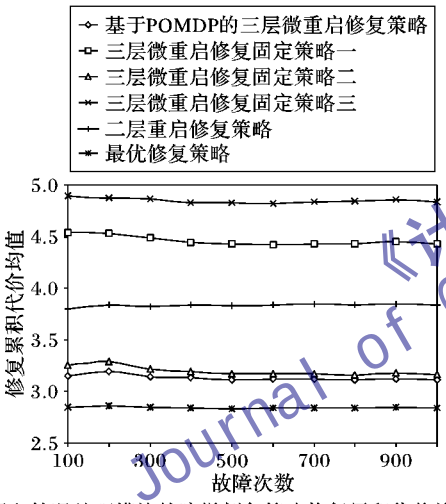

图 7　紧急情况处理模块故障样例各策略修复累积代价均值变化

5　结语

本文设计并实现了基于 POMDP 和三层微重启的 AUV 软件故障自修复方法,通过实验验证了该方法相比传统修复方法有较低的修复开销,并且对于不同故障情况下自修复稳定性较好,有效地解决了在部分可观环境下 AUV 软件故障修复问题。本文使用的 PBVI 算法是在初始信度给定情况下求解最佳修复策略,而真实环境中由于不确定性,各故障发生的初始信度会动态变化,需要动态调整 AUV 软件自修复 POMDP 模型初始信度,对 AUV 软件自修复模型的进一步研究需要在 PBVI 算法基础上引入在线的初始信度调节策略。

参考文献:

[1] XU Y, PANG Y, GAN Y, *et al*. AUV state of the art and prospect [J]. Transactions on Intelligent Systems, 2006, 1(1): 11 – 14. (徐玉如, 庞永杰, 甘永, 等. 智能水下机器人技术展望[J]. 智能系统学报, 2006, 1(1): 11 – 14.)

[2] ALDURGAM M M, DUFFUAA S O. Optimal joint maintenance and operation policies to maximise overall systems effectiveness [J]. International Journal of Production Research, 2013, 51(5): 1319 – 1330.

[3] PSAIER H, SKOPIK F, SCHALL D, *et al*. Behavior monitoring in self-healing service-oriented systems [C]// Computer Software and Applications Conference (COMPSAC). Piscataway: IEEE, 2010: 357 – 366.

[4] ANGELOPOULOS K, SOUZA V E S, PIMENTEL J. Requirements and architectural approaches to adaptive software systems: a comparative study [C]// SEAMS '13: Proceedings of the 8th International Symposium on Software Engineering for Adaptive and Self – Managing Systems. Piscataway: IEEE, 2013: 23 – 32.

[5] WANG Y. System fault quick recovery design based on VxWorks [J]. Application of Electronic Technique, 2008, 34(6): 32 – 34. (王洋. 基于 VxWorks 的系统故障快速恢复设计[J]. 电子技术应用, 2008, 34(6): 32 – 34.)

[6] HU Y, NAN Q, GAO A. Design and implementation of the check pointing and task recovery mechanism based on VxWorks [J]. Journal of Air Force Engineering University: Natural Science Edition, 2013, 14(5): 48 – 52. (胡延苏, 南秦博, 高昂. VxWorks 中任务恢复机制的设计与实现[J]. 空军工程大学学报: 自然科学版, 2013, 14(5): 48 – 52.)

[7] YU K. Dynamic loading mechanism of spaceborne computer technology research based on Reliability[D]. Changsha: National University of Defense Technology, 2009: 1 – 54. (于康. 基于动态加载机制的星载计算机可靠性增强技术研究[D]. 长沙: 国防科学技术大学, 2009: 1 – 54.)

[8] SUN K, MU D, ZHANG H. Design and implementation of high available fault-tolerant system based on VxWorks [J]. Computer Technology and Development, 2012, 22(4): 123 – 125. (孙锴, 慕德俊, 张慧翔. 基于 VxWorks 的高可用容错系统的设计与实现[J]. 计算机技术与发展, 2012, 22(4): 123 – 125.)

[9] CANDEA G, CUTLER J, FOX A. Improving availability with recursive microreboots: a soft-state system case study [J]. Performance Evaluation Journal, 2004, 56(1/2/3): 213 – 248.

[10] LE M, TAMIR Y. Applying microreboot to system software [C]// Proceedings of the 2012 IEEE International Conference on Software Security and Reliability. Piscataway: IEEE, 2012: 1 – 20.

[11] XU Z, ZHAO F, BHAGALIA R, *et al*. Generic rebooting scheme and model-based probabilistic pruning algorithm for tree-like structure tracking [C]// Proceedings of the 2012 9th IEEE International Symposium on Biomedical Imaging. Piscataway: IEEE, 2012: 796 – 799.

[12] ALONSO J, MATIAS R, VICENTE E, *et al*. A comparative evaluation of software rejuvenation strategies [C]// Proceedings of the 2011 Third International Workshop on Software Aging and Rejuvenation. Piscataway: IEEE, 2011: 26 – 31.

[13] WANG Z, GUO C, LIU F, *et al*. Research on the application of artificial neural network in the fine-grained software rejuvenation of computing system [J]. Chinese Journal of Computers, 2008, 31(7): 1268 – 1274. (王湛, 郭成昊, 刘凤玉, 等. 神经网络在计算系统软件抗衰重启技术中的应用研究[J]. 计算机学报, 2008, 31(8): 1268 – 1274.)

[14] YOU J. Research on fine grained software rejuvenation policy and related technology [D]. Nanjing: Nanjing University of Science and Technology, 2006: 1 – 88. (游静. 细粒度软件抗衰策略及相关技术研究[D]. 南京: 南京理工大学, 2006: 1 – 88.)

[15] SHI C, ZHANG R, ZHAO J. Software self-recovery method of AUV based on micro-reboot [C]// Proceedings of the 2011 9th World Congress on Intelligent Control and Automation. Piscataway: IEEE, 2011: 91 – 96.