分布式系统监控资源多目标优化分配

何 盼[1] 袁 月[1] 吴开贵[2]
(中国科学院重庆绿色智能技术研究院 重庆 401122)[1] (重庆大学计算机学院 重庆 400044)[2]

摘 要 针对分布式系统松耦合和动态配置的特点,提出采用监控资源分配进行组件监控周期的选择以达到可靠性保障和资源优化的目的。为了建立监控资源分配模型,首先采用马尔可夫链理论分析了监控策略下的系统可靠性模型;其次分析了监控机制的两种不同代价;再次选择了系统可靠性约束下的多目标监控资源分配模型,它通过最小化监控代价选择恰当的组件监控周期;最后应用遗传算法解决该优化模型。通过实验验证了监控资源分配的必要性和在可靠性优化中的作用,实验表明:监控资源分配能够达到资源优化和可靠性保障的目的;与单目标资源分配相比,多目标分配能够达到更好的优化效果。

关键词 监控资源分配,分布式系统,可靠性优化,马尔可夫链
中图法分类号 TP311.5 **文献标识码** A

Optimal Multi-objective Monitoring Resources Allocation in Distributed Systems

HE Pan[1] YUAN Yue[1] WU Kai-gui[2]
(Chongqing Institute of Green and Intelligent Technology, Chinese Academy of Sciences, Chongqing 401122, China)[1]
(College of Computer Science, Chongqing University, Chongqing 400044, China)[2]

Abstract Against the loosely coupled and dynamic configuration features of distributed systems, multi-objective monitoring resources allocation oriented at optimal monitoring interval selection was studied to achieve the balance between resources optimization and reliability maintenance. To propose the allocation model, first of all, the Markov chain theory was employed to analyze the reliability of systems under monitoring mechanism. Secondly, two different kinds of monitoring resources cost models were analyzed. On the top of that, a multi-objective monitoring resources allocation model was built under the system reliability constraint. This model chooses the appropriate monitoring rate for each component through minimizing the monitoring cost. Finally, a genetic algorithm was used to solve the allocation and optimization model. Experimental studies results prove the necessity and impact of monitoring resources allocation in reliability optimization. The results also show that monitoring resources allocation can help to achieve resources optimization and reliability maintenance. Comparing with single-objective resources allocation, multi-objective resources allocation approaches can get better optimization results.

Keywords Monitoring resources allocation, Distributed system, Reliability optimization, Markov chain

1 引言

分布式系统的组件分布在相互连通的网络中,组件之间通过消息传递进行通信和行为协调。传统的分布式系统利用面向对象的编程方法,采用函数调用或远程函数调用的形式在不同组件间交换信息,属于紧耦合方式。面向服务架构(SOA)具有松耦合的特点,改变了系统的组织方式,目前在分布式系统中应用较为普遍。基于SOA的分布式系统主要由两部分组成:具体服务和服务组合。具体服务提供了特定的功能,是组成系统的最小单元。服务组合采用规范的方法定义了多个服务的关系及执行顺序。服务组合中的服务被称为抽象服务,它仅代表了该服务需要完成的功能,不代表具体执行的服务。在网络环境中,可能存在多个具体服务能够完成一个抽象服务的功能,但其中一个(或多个)会被标注为正在使用的服务,并被实际调用。系统中的组件可能来自于网络中的任何机构,网络中也可能同时存在多个实现相同功能的组件。此外,系统中的组件可以动态地被其它组件所替换。

在系统运行一段时间后,各个组件的可靠性都将会因各种原因衰退,如网络负荷的增加等。为了保障系统的可靠性,为系统组件配置冗余和组件测试是最常用的两种方法。由于分布式系统中的组件来自于网络,不易再通过传统的测试手段来维护和提高其可靠性,因此监控被作为一种常用手段用于系统可靠性的保障中[1]。除了组件的冗余,监控资源的部署也将对系统的可靠性产生一定影响,同时也将给系统带来

到稿日期:2013-09-17 返修日期:2013-11-21 本文受重庆市科技攻关计划项目(cstc2011ggC40008),重庆市自然科学基金项目(CSTC 2011BB2064)资助。

何 盼(1984—),女,博士,助理研究员,主要研究方向为可信计算,E-mail:hepan@cigit.ac.cn;袁 月(1986—),女,硕士,研究实习员,主要研究方向为控制工程;吴开贵(1966—),男,博士,教授,主要研究方向为可信服务计算等。

额外的开销，然而服务监控资源的分配在可靠性优化设计中的作用却很少得到研究。

本文在传统系统可靠性优化设计问题的基础上加入了监控周期的影响，通过分配监控资源对系统进行可靠性优化设计。已有研究是从单目标优化入手对监控资源进行分配[2]，本文则从监控点个数和监控周期两个方面入手进行资源优化分配。本文首先分析了分布式冗余系统中基于监控的组件可靠性模型，建立了基于不同监控点的整体系统可靠性模型。在分析了监控代价的基础上，改进了传统的测试资源模型，形成了监控资源分配模型。最后采用遗传算法解决了该问题。

2 分布式冗余系统监控机制

为了保障分布式系统的可靠性，现有研究采用了多种方式增强组件或系统的可靠性，主要包括以下两类[3]：一种方式是采用传统的冗余机制，为系统中的每个服务配置备份服务[4,5]。在某个服务出错时，系统可以灵活地切换到冗余服务上，这也正是面向服务架构的优点。但是网络环境中的服务都是始终运行的，不能保障冷备份机制的采用，每个备份服务也随时可能被其它系统所使用。而积极冗余机制不能保障在服务失效时备份服务是可用的。一旦备份服务同时出错，系统同样会失效，也将影响系统的性能和可靠性。另一种方法是当组合中某个服务失效时，在网络中重新选择可用的服务并进行替换[6]。该方法能够保障重选的服务都是正常运行的，但服务的选择过程将中断系统的正常执行，如果服务的选择复杂度较高，将严重影响系统的性能。所以为了及时发现组件及其备份组件的运行状态并采取相应措施，常常将监控与动态替换机制应用于分布式系统中。在分布式冗余系统中，已有研究提出基于监控的冗余组件动态配置方法[3]。系统将周期地监控并评估组件的可靠性，如果组件的可靠性低于某个临界值，则将对冗余组件的状态进行检查，一旦冗余组件不可用，就重新选择能够实现该组件功能的其它服务作为冗余组件，该过程如图 1 所示。

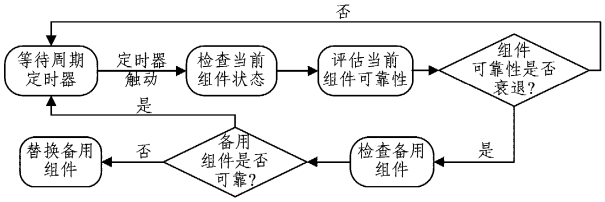

图 1 分布式系统中的组件监控替换机制

3 监控资源分配模型与算法

3.1 监控资源分配模型

3.1.1 组件可靠性模型

基于监控的组件可能具有多种状态，采用传统的 RBD 和 Fault Tree 不能对其可靠性进行较好的分析，所以我们采用马尔可夫链构建基于监控的冗余组件状态变化模型，如图 2(a)所示。状态 2 表示两个组件均可正常运行的状态，状态 1 表示其中一个组件失效的状态，状态 0 表示两个组件均失效的状态，状态 3 表示在一个组件失效时监控器被触发的状态。当监控器判断出系统中仅有一个组件可用时，系统将对失效组件进行重新选择和绑定。状态间的转移主要是通过服务的失效、监控的触发以及服务的替换而产生。我们做如下的假设：①冗余的多个组件的失效率相同；②单个组件的失效周期为 X，且 X 满足指数分布，即 $X\sim\mathrm{EXP}(\lambda)$；③当监控器发现其中一个冗余组件可靠性降低时，进行组件重选的时间 Y 满足指数分布，即 $Y\sim\mathrm{EXP}(\mu_m)$。

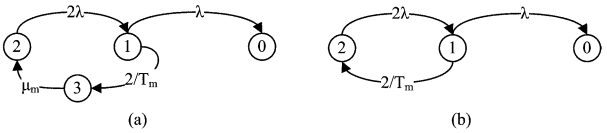

图 2 基于监控的冗余组件状态变化模型

假设监控等待的周期为 Z_m，即从状态 1 转移至状态 3 所需的时间为 Z_m。由于采用轮询监控的方式，假设监控周期为 T_m，则 Z_m 的取值是$[0,T_m]$区间中的任意随机数。假设 Z_m 服从于参数为 $2/T_m$ 的指数分布，即 $Z_m\sim\mathrm{EXP}(2/T_m)$，如图 2(a)所示。为了方便计算，令 $\lambda_m=1/T_m$。假设组件从状态 1 转移至状态 2 所需的时间为 Z_m'，其为随机变量 Z_m 与 Y 之和。由于 Z_m 与 Y 均满足指数分布，因此 Z_m'满足 2 阶亚指数分布：$Z_m'\sim\mathrm{HYPO}(\lambda_m,\mu_m)$，其分布函数如式(1)所示。

$$F_{Z_m'}(t)=\frac{\lambda_m\mu_m}{\mu_m-\lambda_m}(\mathrm{e}^{-\lambda_m t}-\mathrm{e}^{-\mu_m t}) \tag{1}$$

假设在实际运行中，监控的周期远大于组件进行判断和重选的时间，可以对 Z_m'过程进行简化，简化后的状态转移图如图 2(b)所示。易知上述模型满足连续时间齐次马尔可夫链。令 Z 表示该马尔可夫过程，Z 的性质主要由以下两个参数决定：系统的初始状态和状态转移概率。令 $\pi_i(t)$表示系统在 t 时刻处于状态 i 的概率，如式(2)所示。

$$\pi_i(t)=P(Z(t)=i),i=0,1,2 \tag{2}$$

$\pi_i(0)$表示初始时刻系统处于各状态的概率，图 2(a)所示的模型中，$\pi_i(0)$满足如下关系：

$$\pi_i(0)=\begin{cases}1, & i=2\\ 0, & i=0,1\end{cases} \tag{3}$$

令 $p_{ij}(v,t)$表示 v 时刻系统处于状态 i 时，在 t 时刻转移到状态 j 的概率，如式(4)所示，$q_{ij}(t)$表示 t 时刻系统从状态 i 转移至状态 j 的频率，如式(5)所示。

$$p_{ij}(v,t)=P(Z(t)=j\,|\,Z(v)=i) \tag{4}$$

$$q_{ij}(t)=-\frac{\partial}{\partial t}p_{ij}(v,t)\,|_{v=t}=\lim_{h\to 0}\frac{p_{ij}(t,t+h)}{h} \tag{5}$$

对于图 2(b)所示的齐次马尔可夫链，$p_{ij}(v,t)$与时间 v 的变化无关，只与时间的差值 $t-v$ 有关，所以可以证明在齐次马尔可夫链中，$q_{ij}(t)$的变化与时间 t 无关，记为 q_{ij}，可参见图 2(b)。通过柯尔莫哥洛夫向前方程，根据 $p_{ij}(v,t)$和 $\pi_i(0)$可以求得 t 时刻系统的瞬时概率 $\pi_i(t)$：

$$\frac{\partial\pi_i(t)}{\partial t}=\sum_{k=0}^{2}\pi_k(v,t)q_{ki} \tag{6}$$

采用 laplace 变换和逆 laplace 变换可以获得 $\pi_i(t)$。假设 $R(t,\lambda_m)$表示时刻 t、监控频率为 $1/\lambda_m$ 的组件可靠性，只有状态 0 表示组件的失效状态，则该组件的瞬时可靠性 $R(t,\lambda_m)=1-\pi_0(t)$，如式(7)所示。

$$R(t,\lambda_m)=-\lambda\left(\frac{4\lambda\mathrm{e}^{\frac{(-a+b)t}{2}}}{b(-a+b)}+\frac{4\lambda\mathrm{e}^{\frac{-(a+b)t}{2}}}{b(a+b)}\right) \tag{7}$$

其中，$a=2\lambda_m+3\lambda$，$b=\sqrt{4\lambda_m^2+12\lambda\lambda_m+\lambda^2}$。

3.1.2 分布式系统可靠性模型

与传统软件或硬件系统相同，分布式系统的可靠性主要

由两方面决定：系统中每个组件的可靠性及系统的结构。采用 RBD 及 FT 方法，系统的可靠性可由通过各种结构组织的组件可靠性计算得到。对具有固定结构的系统如串并联系统、星型系统等，已有较多的文献对其可靠性的分析做出研究，并有相当成熟的结论[7]。针对具有复杂网状结构的系统，不少学者采用马尔可夫链的相关理论，建立了基于状态的系统可靠性分析理论。文献[8]提出了基于体系结构的系统可靠性分析方法，其利用系统的组件状态转换图构建 DTMC 并用于计算系统可靠性，是目前较为通用的一种方法[9]。假设系统内部采用了层次方法，R_i 表示系统中第 i 个组件的可靠性，V_i 表示第 i 个组件的平均访问次数，m 表示组件个数。采用基于体系结构的分析方法，系统可靠性如式(8)所示。

$$R_{sys}=\prod_{i=1}^{m}R_i^{V_i} \tag{8}$$

在不考虑网络传输可靠性的情况下，分布式系统与传统系统的差别在于分布式的组件间存在多种执行关系，包括串行、并行、选择循环等。分布式系统允许组件的并行执行，这在传统系统中通常是不存在的，现有研究中已提出了多种方法[10]对系统结构进行转换并可以采用式(8)所示的模型加以解决。

3.1.3 监控代价模型

监控资源的配置可能造成多种代价，如存储资源的占用，计算资源的耗费等。在基于轮询监控的分布式冗余系统中主要有两类代价：一类是监控点的配置所造成的代价，另一类是定时监控措施所造成的代价。

监控点配置的总代价是系统中所有监控点配置代价之和[2]。假设第 i 个组件设置监控点的代价为 C_i，监控频率为 λ_{m_i}，系统总代价 C_{sys} 如式(9)所示。监控措施的代价主要与每个组件的监控周期相关。系统监控的总代价是系统中所有组件监控代价之和[2]。假设系统总代价为 M_{sys}，组件 i 配置监控的单元时间代价为 M_i，则系统监控的总代价如式(10)所示。

$$C_{sys}=\sum_{i=1}^{m}p_iC_i \tag{9}$$

其中，$p_i=\begin{cases}1, & \lambda_{m_i}>0\\ 0, & \lambda_{m_i}=0\end{cases}$

$$M_{sys}=\sum_{i=1}^{m}\lambda_{m_i}M_i \tag{10}$$

3.1.4 监控资源分配模型

在分析了系统可靠性、两种代价的基础上，本节拟解决在可靠性约束的条件下对系统内组件的监控周期进行分配的问题，拟达到系统的可靠性的保障及监控周期的最小化。假设系统中对于每个组件配置监控的代价 C_i 和 M_i 是相同的，该最优分配问题的模型如式(11)所示。

$$\text{Minimize:}C_{sys}=\sum_{i=1}^{m}p_i,\text{其中 }p_i=\begin{cases}1, & \lambda_{m_i}>0\\ 0, & \lambda_{m_i}=0\end{cases}$$

$$\text{Minimize:}M_{sys}=\sum_{i=1}^{m}\lambda_{m_i} \tag{11}$$

$$\text{s.t.}\quad R_{sys}=\prod_{i=1}^{m}R_i^{V_i}\geqslant R_0,\text{其中 }R_i=R(t,\lambda_{m_i})。$$

3.2 监控资源分配算法

在式(11)所示的问题中，两个目标函数是监控频率的多元线性函数，但约束函数可靠性是监控频率的多元非线性函数，所以该问题属于非线性规划问题。虽然采用线性规划的方法可以取得该问题的近似解，但该解的效果还有待改进。同时，线性规划方法只能按一定规则将该问题转化为单目标优化问题进行解决，而很难取得两个目标之间的平衡解集。所以在本节中，我们拟采用进化算法对该问题的近似最优解进行搜索。同时在进化算法中，研究者已提出有多种多目标优化方式，其中第二代快速非支配排序遗传算法(NSGA II)在多目标选择策略方面比其它多目标算法略胜一筹[11]，所以本章中拟采用遗传算法进行最优解的搜索并采用第二代非支配排序方法进行多目标解的选择。算法主要包含以下步骤：①首先采用模拟二级制交叉算子和多项式变异算子进行交叉变异操作；②然后采用基于随机漫步的局部搜索策略进行局部解优化；③最后采用非支配排序对两个目标进行全局解排序。算法框架的伪代码如下：

①设置父种群 $P_0=\emptyset$，子种群 $Q=\emptyset$，进化代数 $t=0$。

②随机生成 N 个初始个体，对父种群 P_0 进行初始化：$P_0=[p_1,p_2,\cdots,p_N]$，每个 1 维数组 p_i 是随机生成的 m 个组件的监控频率 λ_{m_i}：$p_i=(\lambda_{m_1},\lambda_{m_2},\cdots,\lambda_{m_m})$。

③while 进化代数 t＜最大进化代数 t_{max}时，执行以下步骤：

1)for i=1 到 N

a. if rand()＜交叉率，随机选择两个个体，采用交叉算子产生两个新个体 x 与 x′，并将其放入子种群 Q 中。

b. if rand()＜变异率，随机选择一个个体，采用变异算子产生一个新个体 x″，并将其放入子种群 Q 中。

c. if rand()＜局部搜索率，随机选择一个个体，采用基于随机漫步的局部搜索方法随机搜索其近邻，并将近邻中的可行解放入子种群 Q 中。

2)end for

3)合并父种群 P_t 和子种群 Q 生成新的种群 R。

4)计算种群 R 中各个体产生的约束值和目标值。

5)计算 R 中各个体的非支配字体和聚集距离，并通过该值对个体进行倒序排列。

6)选择 R 中前 N 个个体，并放入下一代父种群 P_{t+1}中。

7)t=t+1。

④ end while

4 实验

在本节中，我们采用了文献[12]中采用的一个实例系统进行实验。该系统的组织结构图如图 3 所示。该系统的组件间包含了基本的 4 种连接关系，即顺序执行，选择执行，循环执行，并行执行。系统中各组件之间的转移概率可参见文献[12]。

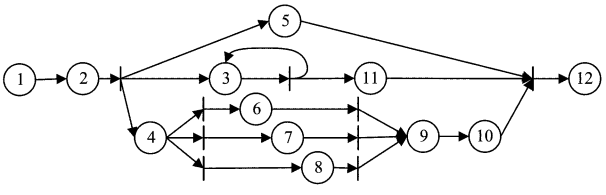

图 3 系统示例

本实验中主要采用了 Membrane 数据集[13]。Membrane 是一个开源的 Web 服务注册与监控网站。Membrane 会对其中注册的服务进行定期的监控和评估，该网站中提供了服务的 MTTF 和 MTTR 等信息，可以用于本实验中。为了测试算法的有效性，我们在真实服务的可靠性数据集中随机选择了 10 组每个组件的可靠性数据：每个组件的失效率 λ_i。在

MOMA 算法的初始化过程中，每个组件的监控率 λ_{m_i} 被设为范围为(0,1)、单位为 1/min 的随机数。所以每个组件的最小监控周期为 1min，最大监控周期为∞(此时表示组件不需要设置监控器)。在 MOMA 算法中，其它控制参数的设定如表 1 所列。

表 1　MOMA 算法的控制参数值设定

算法总迭代次数 t_{max}	每代个体数	交叉率	变异率	局部搜索率	局部搜索迭代次数	局部搜索内部迭代次数
200	100	0.9	0.1	0.1	10	5

4.1　监控频率对系统可靠性的影响

在本实验中，我们主要分析不同组件的监控频率变化对整体系统可靠性的影响程度。首先根据图 3 所示的系统结构图构建相应的系统状态转移模型，可以得到系统中每个组件 i 的平均访问次数 $V_i=[1,1,3.03,1,1,0.7,0.2,0.1,1,1,1]$。在原始状态下，系统中无监控器配置，此时系统可靠性为 $R_{sys}\approx R_1R_2R_3^{3.03}R_4R_5R_6^{0.7}R_7^{0.2}R_8^{0.1}R_9R_{10}R_{11}R_{12}$。在本实验中，我们采用了一组组件可靠性参数进行实验，该组数据如表 2 所列。

表 2　案例分析示例数据

序号	λ(1/min)	序号	λ(1/min)	序号	λ(1/min)	序号	λ(1/min)
1	0.00022714	4	0.00029817	7	0.00045451	10	0.00018154
2	0.00027192	5	0.0006788	8	0.00047322	11	0.00067923
3	0.00044514	6	0.00036196	9	0.000669	12	0.00048555

假设时刻 $t=10$h 时，不同的组件监控频率的变化对整体系统可靠性的影响如图 4 所示。图 4(a)表示了不同组件的监控频率变化对单个组件可靠性的影响程度，图 4(b)表示了不同组件的监控频率变化对整体系统可靠性的影响程度。

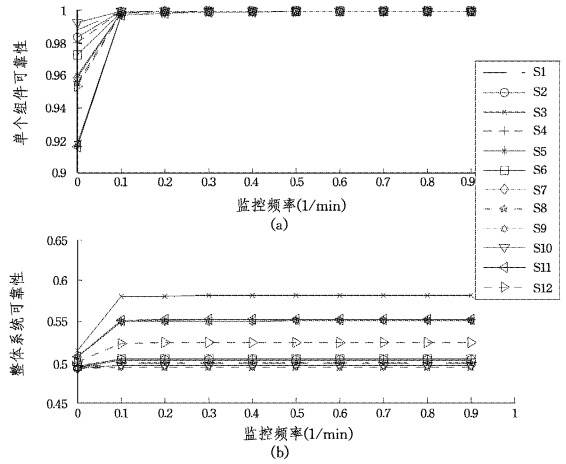

图 4　不同组件上的监控频率变化对可靠性的影响

从图 4 中，我们可以得出以下结论：①随着每个组件监控频率的增加(即监控周期的缩短)，每个组件的可靠性和整体系统的可靠性都有所增加，并且增加的幅度不同。②随着监控频率的增加，组件可靠性的增长幅度越来越小。过高的监控频率反而不能促进可靠性的增长，所以对各个组件的监控频率进行优化分配是有必要的。③增加单个组件的监控频率有助于增加系统可靠性，但最终系统可靠性的增加会停止，并接近某固定值(例如，0.6)。所以为了确保系统可靠性达到一定标准，需要对系统中多个组件设置监控。

4.2　案例分析

在本实验中，我们采用遗传算法，针对系统可靠性约束进行监控频率的选取。本实验中采用的参数与上一个实验相同(见表 2)。假设 $t=10$h 时，在监控资源分配前，系统可靠性为 0.4902。假设该时刻系统可靠性需要被增加到 0.90 或 0.99，对每个组件的监控频率进行计算可以得到相应的监控配置结果，如表 3 所列。由于算法会返回一组监控点与监控频率之和的非支配解集，在该组解集中，我们选取了监控点和监控频率之和最少的解作为说明。

表 3　案例分析示例结果

组件序号	λ_m(1/min)			
	t=10h，R_0=0.90		t=10h，R_0=0.99	
	监控点最少	监控频率之和最少	监控点最少	监控频率之和最少
1	0.0000	0.0002	0.2360	0.0736
2	0.0000	0.0110	0.2523	0.0784
3	0.1335	0.0261	0.6552	0.2099
4	0.0000	0.0113	0.3244	0.0872
5	0.2126	0.0194	0.3953	0.1947
6	0.0698	0.0146	0.3327	0.0961
7	0.0000	0.0014	0.1450	0.0525
8	0.0000	0.0000	0.0000	0.0415
9	0.1055	0.0196	0.4251	0.2069
10	0.0000	0.0012	0.1647	0.0469
11	0.1728	0.0197	0.4355	0.2018
12	0.1267	0.0247	0.4454	0.1461
C_{sys}	6	11	11	12
M_{sys}	0.8208	0.1493	3.8118	1.4356

表 3 所列的数据表明：①增加监控点的个数可以减少每个监控点的监控频率，反之亦然；②同一时刻当系统的可靠性需求增加时，需要为系统中更多的组件配置监控策略或提高监控的频率。

4.3　与单目标优化方法比较

为了体现多目标分配的优点，在本实验中我们将多目标分配结果与单目标优化方法的分配结果进行比较。在以往研究中采用了基于敏感度分析的贪婪算法进行监控资源分配及可靠性优化[2]。针对每种输入值，该方法仅返回一组解，而多目标算法将返回一个非支配解集，集合中的每个解都互为非支配解。

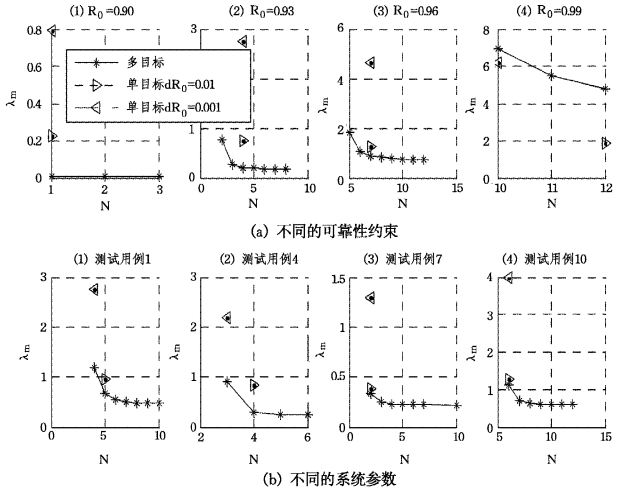

图 5　多目标优化方法与单目标优化方法的结果比较

首先我们假设系统中单个组件的可靠性确定如表 2 所列，$t=10$h。系统的可靠性约束由 0.90 向 0.99 增加。图 5(a)展示 4 组测试案例下，多目标算法得到的非支配集和单目标算法的解。其中 dR_0 表示单目标算法中每个组件最低

(下转第 77 页)

and Software,1999,47(2):173-181
[11] Yoo S,Harman M. Pareto efficient multi-objective test case selection[C]//Proceedings of the 2007 international symposium on Software testing and analysis. ACM,2007:140-150
[12] 程俊,李征,赵瑞莲. CPU+GPU 异构并行多目标测试用例预优化技术[C]//第七届中国测试学术会议(CTC2012). 2012,6:116-121
[13] Malz C,Jazdi N,Gohner P. Prioritization of Test Cases Using Software Agents and Fuzzy Logic[C]//2012 IEEE Fifth International Conference on Software Testing,Verification and Validation (ICST). IEEE,2012:483-486
[14] Tonella P, Avesani P, Susi A. Using the case-based ranking methodology for test case prioritization[C]//22nd IEEE International Conference on Software Maintenance,2006(ICSM'06). IEEE,2006:123-133
[15] Kayes M I. Test case prioritization for regression testing based on fault dependency[C]//2011 3rd International Conference on Electronics Computer Technology (ICECT). IEEE,2011,5:48-52
[16] Deep K,Chauhan P,Pant M. Totally disturbed chaotic Particle Swarm Optimization[C]//2012 IEEE Congress on Evolutionary Computation (CEC). IEEE,2012:1-8
[17] 廖子贞,罗可,周飞红,等. 一种自适应惯性权重的并行粒子群聚类算法[J]. 计算机工程与应用,2007,43(28):166-168
[18] Lian Z,Gu X,Jiao B. A similar particle swarm optimization algorithm for permutation flowshop scheduling to minimize makespan[J]. Applied Mathematics and Computation,2006,175(1):773-785
[19] Hla K H S,Choi Y S,Park J S. Applying particle swarm optimization to prioritizing test cases for embedded real time software retesting[C]//IEEE 8th International Conference on Computer and Information Technology Workshops,2008(CIT Workshops 2008). IEEE,2008:527-532
[20] 沈恺涛,胡德敏. 基于云计算和改进离散粒子群的任务调度研究[J]. 计算机测量与控制,2012,20(011):3070-3072

(上接第 67 页)

可靠性增长幅度,N 表示监控点数目,λ_m 表示监控频率之和。其次,我们假设系统的可靠性约束不变,为 0.95,t=10h,采用了 10 组随机选择的组件可靠性参数数据。同理,单目标算法只能获得一组解,并且被包含于 MOMA 方法获得的非支配集中。其中 4 组数据获得的详细结果如图 5(b)所示。根据图 5 可以看出,单目标的解被包含在多目标的非支配集中,并且多目标方法提供了更多种监控配置方式选择的方案。虽然部分情况下单目标方法看似比 MOMA 的解更好,但在此情况下单目标方法的解达不到可靠性约束条件要求。

结束语 由于分布式系统中的组件来自于网络,不易再通过传统的测试手段来维护和提高其可靠性,因此监控被作为一种常用手段用于系统可靠性的保障中。除了组件的冗余,监控也将对系统的可靠性产生一定影响,同时也将给系统带来额外的开销,然而服务监控在可靠性分配中的作用却很少得到研究。本文在传统面向可靠性优化设计的资源分配问题的基础上加入了监控机制的影响,通过分配监控资源对系统可靠性进行优化。首先分析了分布式冗余系统中基于监控的组件可靠性模型,建立了基于不同监控点的整体系统可靠性模型。在分析监控代价的基础上,改进了传统的测试资源模型,形成了监控资源分配模型,提出在可靠性约束条件下将监控点的数目和监控频率之和作为系统可靠性优化设计的两大资源优化目标。然后采用遗传算法解决了该问题。为了分析监控资源分配对系统可靠性的影响,本文分别对单目标和多目标监控资源分配进行了实验,实验结果表明:①随着可靠性约束的提高,需要增加系统中被监控的组件数目或者提高组件监控频率。②与单目标分配相比,多目标分配方法能够提供更好的资源优化方案。

参 考 文 献

[1] Ben H R,Emna F,Khalil D,et al. A large-scale monitoring and measurement campaign for Web Services-based applications [J]. Concurrency Computation Practice and Experience,2010,22(10):1207-1222
[2] He Pan,Wu Kai-gui,Wen Jun-hao, et al. Monitoring resources allocation for service composition under different monitoring mechanisms[C]//Proceedings of 5th International Conference on Complex,Intelligent and Software Intensive Systems,2011. Washington,D C:IEEE Computer Society,2011:263-270
[3] Dai Yu, Yang Lei, Zhang Bin. QoS-driven self-healing Web service composition based on performance prediction [J]. Journal of Computer Science and Technology,2009,24(2):250-261
[4] Yu Tao, Lin K-J. Adaptive algorithms for finding replacement services in autonomic distributed business processes[C]//Proceedings of International Symposium on Autonomous Decentralized Systems,2005. Washington,D C:IEEE Computer Society,2005:427-434
[5] Girish C,Koustuv D,Arun K,et al. Adaptation in Web Service composition and execution[C]//Proceedings of IEEE International Conference on Web Services, 2006. Washington, D C: IEEE Computer Society,2006:549-557
[6] Gerardo C, Di P M, Raffaele E, et al. QoS-aware replanning of composite Web services[C]//Proceedings of IEEE International Conference on Web Services, 2005. Washington, D C: IEEE Computer Society,2005:121-129
[7] Rüdiger R. Reliability analysis—a review and some perspectives [J]. Structural Safety,2001,23(4):365-395
[8] Gokhale S S, Trivedi K S. Analytical models for architecture-based software reliability prediction: A unification framework [J]. IEEE Transactions on Reliability,2006,55(4):578-590
[9] Gokhale S S. Architecture-based software reliability analysis: Overview and limitations [J]. IEEE Transactions on Dependable and Secure Computing,2007,4(1):32-40
[10] Wang Li-jun,Bai Xiao-ying,Zhou Li-zhu,et al. A hierarchical reliability model of service-based software system[C]//Proceedings of 33rd Annual IEEE International Computer Software and Applications Conference, 2009. Washington, D C: IEEE Computer Society,2009:199-208
[11] Wang Zai,Tang Ke,Yao Xin. Multi-objective approaches to optimal testing resource allocation in modular software systems [J]. IEEE Transactions on Reliability,2010,59(3):563-575
[12] Xia Yun-ni,Wang Hanp,Huang Y,et al. A stochastic model for workflow QoS evaluation [J]. Scientific Programming,2006,14(3/4):251-265
[13] Directory of Public Soap Web Services[EB/OL]. http://www.service-repository.com