基于模糊 Petri 网的 Web 性能优化建议推理*)

叶新铭　王玉龙　李秀华
（内蒙古大学计算机学院　呼和浩特 010021）

摘　要　从 Web 应用性能状况得到性能瓶颈及解决方案是 Web 性能测试报告的主要功能。然而，当前流行的 Web 性能测试工具仅仅能够提供被测 Web 应用的性能状况，性能瓶颈的确定和解决方案的选择均需要测试人员手动完成。根据 Web 性能数据自动确定其瓶颈所在和解决方案将显著提高测试人员的工作效率。本文提出基于模糊 Petri 网的 Web 性能优化建议推理方法，使测试报告能够提供更有价值的信息。

关键词　Web，性能，模糊 Petri 网，推理

Web Performance Optimization Suggestion Reasoning Based on Fuzzy Petri Net

YE Xin-Ming　WANG Yu-Long　LI Xiou-Hua
(College of Computer Science, Inner Mongolia University, Huhhot 010021)

Abstract　Most popular Web performance testing tools can provide performance evaluation in their testing report, but few of them are able to show performance bottleneck and no one indicate solutions on upgrade Web applications being tested. This paper bring forwards a new reasoning method based on Fuzzy Petri Net, which automatically produce suggestions for optimizing Web system performance based on the testing data it gathers.

Keywords　Web, Performance, Fuzzy Petri net, Reasoning

1　引言

中国互联网络信息中心 1997 年至 2006 年“中国互联网络发展状况统计报告”[1]显示，10 年间我国 Web 站点增幅为 461.8%。虽然网络带宽也在同时增长，但是 Web 访问速度慢始终占据用户对 Internet 不满之处的前 3 名。Web 性能测试工具的成熟和日渐广泛应用是解决 Web 性能差这一难题的契机。

目前，成熟的 Web 性能测试工具共有 47 个[2]。这些工具虽然在易用性、多协议支持、测试数据准确性等方面已达到较高水平，但是在测试报告的智能化上均显不足。测试结果的分析基本上完全依赖测试人员的能力和经验。

测试过程的自动化结合优化策略的自动生成才能真正大幅提高 Web 性能测试的效率，从而有效解决 Web 访问速度慢的难题。

2　模糊 Petri 网推理

定义 1(Petri 网结构)　一个 Petri 网结构 C 是一个四元组，$C=(P,T,I,O)$。$P=\{p_1,p_2,\cdots,p_n\}$ 是一个库所的有限集，$n\geqslant 0$。$T=\{t_1,t_2,\cdots,t_m\}$ 是一个变迁的有限集，$m\geqslant 0$。库所集和变迁集不相交，$P\cap T=\phi$。$I:T\rightarrow P^{\infty}$ 是输入函数，它将变迁映射成库所的袋子(bag)。$O:T\rightarrow P^{\infty}$ 是输出函数，它将变迁映射成库所的袋子。

定义 2(Petri 网图)　一个 Petri 网图 G 是一个有向连通图，$G=(V,A)$，其中 $V=\{v_1,v_2,\cdots,v_s\}$ 是一个顶点的集合，$A=\{a_1,a_2,\cdots,a_r\}$ 是一个有向弧的袋子，$a_i=(v_j,v_k)$，且 $v_j,v_k\in V$。集合 V 可以分成两个互不相交的集合 P 和 T，即 $V=P\cup T$，$P\cap T=\phi$，并且对于每一个有向弧，$a_i\in A$，如果 $a_i=(v_j,v_k)$ 则要么 $v_j\in P$ 且 $v_k\in T$ 要么 $v_j\in T$ 且 $v_k\in P$[3]。

定义 3(模糊 Petri 网)　一个模糊 Petri 网是一个六元组 $FPN(P,T,I,O,X,Y)$。其中 P,T,I,O 的含义与 Petri 网结构的含义相同。X 为一个映射：$T\rightarrow[0,1]X[t_j]=x_j$ 为推理规则的置信度。Y 为一个映射：$P\rightarrow[0,1]Y[p_i]=y_i$ 为命题的置信度[4]。

带有置信度的推理规则的形式为：IF P_i THEN P_j($CF=\alpha_k$)。这里 P_i 和 P_j 是包含模糊变量的命题，每个命题的真值在 0 到 1 之间。α_k 为规则可信度，$CF\in[0,1]$。它的取值大小表示规则的可信程度。

设 TS 为置信度阈值。本文规定 CF ≥ TS，则结论成立。

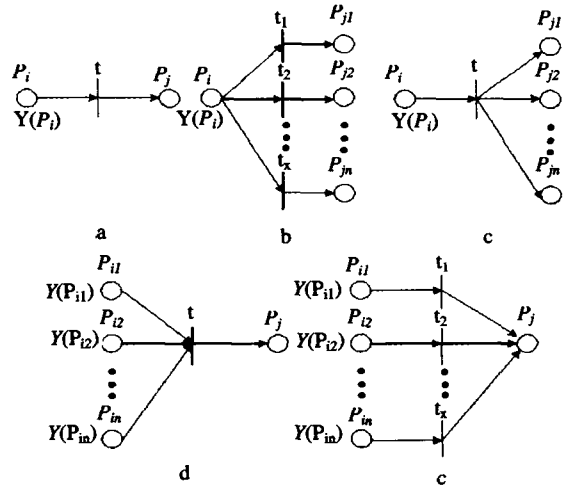

图 1　模糊 Petri 网表示的推理过程

*)基金项目：国家自然基金项目(60563004)，内蒙古科技攻关项目(2002061002)。叶新铭　教授，博士生导师，主要研究方向：计算机网络。王玉龙　硕士研究生；李秀华　硕士研究生，主要研究方向：计算机网络。

在模糊 Petri 网中，用变迁 t 代替每个推理规则，其输入为规则的右部(条件命题)，输出为规则的左部(结论命题)，即推理规则中每个命题对应一个库所 P。当把 $Y(P_i)$加在模糊 Petri 网中时，称之为一个带标识的模糊 Petri 网，其中 $Y(P_i)$称为 Token。

图 1a：表示 IF p_i THEN $p_j(CF=\alpha_k)$；b：IF P_i THEN $p_{j1} \vee p_{j2} \vee \cdots \vee p_{jn}(CF=\alpha_k)$；c：IF p_i THEN $p_{j1} \wedge p_{j2} \wedge \cdots \wedge p_{jn}(CF=\alpha_k)$；d：IF $p_{i1} \wedge p_{i2} \wedge \cdots \wedge p_{in}$ THEN $p_j(CF=\alpha_k)$；e：IF $p_{i1} \vee p_{i2} \vee \cdots \vee p_{in}$ THEN $p_j(CF=\alpha_k)$。

结论命题能否作为一个优化建议取决于该结论命题的置信度是否达到 TS。结论命题的置信度由条件命题置信度和推理规则置信度及具体的推理规则来决定。

a：$Y(p_j)=Y(p_i)\times X(t)$；b：$Y(p_{j1})=Y(p_i)\times X(t_1),\cdots,Y(p_{jn})=Y(p_i)\times X(t_n)$；c：$Y(p_{j1})=Y(p_i)\times X(t)$，$\cdots,Y(p_{jn})=Y(p_i)\times X(t)$；d：$Y(p_j)=\min\{Y(p_{i1}),Y(p_{i2}),\cdots,Y(p_{in})\}\times X(t)$；e：$Y(p_j)=\max\{Y(p_{i1})\times X(t_1),Y(p_{i2})\times X(t_2),\cdots,Y(p_{in})=X(t_n)\}$。

3 Web 优化建议推理过程

3.1 推理规则的建立

定义 4(Web 事务) 一个 Web 事务 $TR(H,W)$是一个 2 元组。其中 H 是 HTTP 请求的集合，W 是客户端等待时间的和。

因为 HTTP 请求不易区分(例如不能仅仅依靠 URL 做区分)，所以本文使用 TR 作为优化处理变量的最小单位。根据我们开发的 Web 性能测试工具 WebTest 能够测量的性能数据，以及作者所查阅和总结的影响 Web 系统性能的因素及解决方法，整理出如下 Web 性能优化建议推理规则：

规则 1 IF TR_i中 H 响应时间大于所有 TR 中 H 响应时间的平均值 THEN TR_i需要优化。

规则 2 IF TR_i需要优化 ∧ TR_i中的 H 响应字节数大于所有 TR 中 H 字节数的平均值 THEN TR_i需要减少文本内容。

规则 3 IF TR_i需要优化 ∧ TR_i 包含资源数大于所有 TR 包含资源数的平均值 THEN TR_i需要减少包含的资源数。

规则 4 IF TR_i需要优化 ∧ TR_i 中 H 响应字节数小于所有 TR 中 H 字节数的平均值 THEN TR_i需要优化服务端脚本代码。

规则 5 IF TR_i需要优化 ∧ TR_i 包含资源的字节数大于所有 TR 包含资源字节数的平均值 THEN TR_i需要降低资源字节数。

规则 6 IF TR_i需要优化服务端脚本代码 ∧ TR_i服务端代码需要访问数据库 THEN 优化 TR_i 访问的数据库对象 ∨ 优化数据库连接池。

规则 7 IF TR_i需要优化服务端脚本代码 ∧ TR_i服务端代码需要访问文件系统 THEN 更新磁盘 I/O 系统。

规则 8 IF TR_i需要减少包含的资源数 ∧ 图片资源的大小均不超过 12kB THEN 将图片资源合并。

规则 9 IF TR_i需要降低资源字节数 ∧ 图片资源的大小超过 12kB THEN 改变图片资源的存储格式 ∨ 降低图片质量。

3.2 推理规则的模糊 Petri 网表示

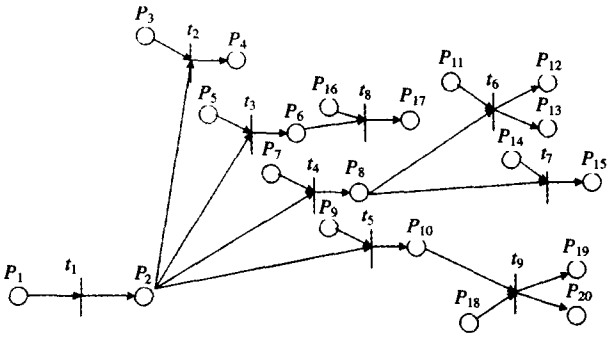

图 2 推理规则的模糊 Petri 网表示

P_1：$=ResponseTime_{TR_i}>Avg(ResponseTime_{TR})$

$$Y(P_1)=\frac{ResponseTime_{TR_i}-Avg(Re\ sponseTime_{TR})}{ResponseTime_{TR_i}}$$

$X(t_1)=1$

P_2：$=TR_i$ 需要优化

P_3：$=ResponseSize_{TR_i}>Avg(Re\ sponseSize_{TR})$

$$Y(P_3)=\frac{Re\ sponse\ Size_{TR_i}-Avg(ResponseSize_{TR})}{Response\ Size_{TRi}}$$

$X(t_2)=0.7$

P_4：$=TR_i$ 需要减少网页文本内容

P_5：$=Re sponseCount_{TR_i}>Avg(Re\ sourceCount_{TR})$

$$Y(P_3)=\frac{ResponseCount_{TR_i}-Avg(Re\ sponseCount_{TR})}{Re\ sponseCount_{TR_i}}$$

$X(t_3)=0.8$

P_6：$=TR_i$ 需要减少网页包含资源的数量

P_7：$=Re\ sponse\ Size_{TR_i}<Avg(Response\ Size_{TR})$

$$Y(P_7)=\frac{Avg(Re\ sponseSize_{TR})-Re\ sponseSize_{TR_i}}{Avg(ResponseSize_{TR})}$$

$X(t_4)=0.9$

P_8：$=TR_i$ 需要优化服务端代码

P_9：$=Re\ source\ Size_{TR_i}>Avg(Resource\ Size_{TR})$

$$Y(P_9)=\frac{Re\ sponse\ Size_{TR_i}-Avg(Re\ sponse\ Size_{TR})}{Re\ sponse\ Size_{TR_i}}$$

$X(t_5)=0.9$

P_{10}：$=TR_i$ 需要减小所包含资源的大小

P_{11}：$=TR_i$ 服务端代码需要访问数据库

$Y(P_{11})=1$

$X(t_6)=0.9$

P_{12}：$=$需要优化 TR_i 访问的数据库对象

P_{13}：$=$需要优化数据库连接池

P_{14}：$=TR_i$ 服务端代码需要访问文件系统

$Y(P_{14})=1$

$X(t_7)=0.9$

P_{15}：$=$需化升级磁盘 I/O 系统

P_{16}：$=Picture\ Re\ source\ Size_{TR}<12K$

$$Y(P_{16})=\frac{12\times1024-Picture\ Re\ sourceSize_{TR}}{12\times1024}$$

$X(t_8)=0.7$

P_{17}：$=$需要合并 TR_i 包含的图片资源

P_{18}：$=Picture\ Re\ source\ Size_{TR_i}>12K$

$$Y(P_{18})=\frac{Picture\ Re\ source\ Size_{TR}-12\times1024}{Picture\ Re\ source\ Size_{TR}}$$

$X(t_9)=0.9$

P_{19}：$=$需要降低 TR_i 包含图片的质量

P_{20}：$=$改变 TR_i 包含图片的存储格式

3.3 推理实现

3.3.1 核心数据结构

推理过程中使用 3 个核心类：代表模糊 Petri 网的 CFuzzyPetriNet、代表库所的 CPlace 和代表变迁的 CTransit。

3.3.2 核心算法

(1)模糊 Petri 网的初始化。在开始推理之前，需要根据事前设置的推理规则构造等价的模糊 Petri 网。为了使推理过程不依赖于某一套特定的推理规则，本文采用从推理规则

文件读入推理规则的方法来初始化模糊 Petri 网。

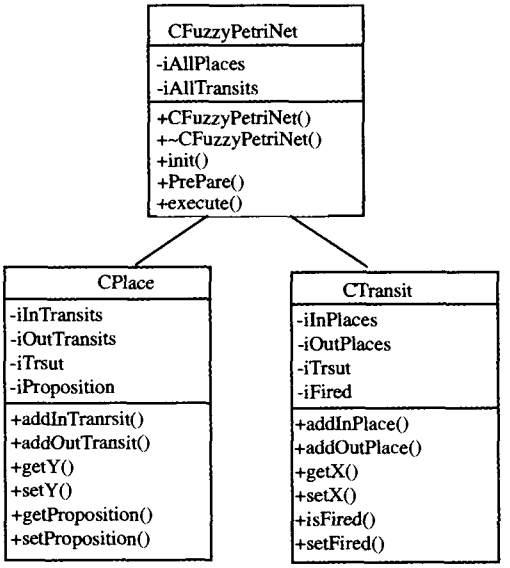

图 3　模糊 Petri 网推理核心类

初始化过程如下：

第 1 步，读入推理规则 XML 文件；

第 2 步，根据推理规则文件构造所有的库所（命题）对象，存入 iAllPlaces 变量中；

第 3 步，根据推理规则文件构造所有的变迁（推理）对象，存入 iAllTransits 变量中；

第 4 步，根据 XML 中每个库所元素的参数添加库所对象的入变迁和出变迁；

第 5 步，根据 XML 中每个变迁元素的参数添加变迁对象的入库所和出库所；

（2）推理准备。虽然对于所有的被测网页推理规则都是相同的，但是由于每一个网页的性能测试结果数据不同，所以在每次推理过程进行之前，需要根据网页的性能测试数据和命题的置信度计算公式来计算所有初始命题的置信度，并将其赋值给模糊 Petri 网对应的库所。其过程如下：

第 1 步，取得第一个库所；

第 2 步，如果该库所没有入变迁，则计算并设置其置信度，否则设置其置信度为 0；

第 3 步，取得下一个库所，如果取到，则转第 2 步；

第 4 步，结束，返回。

（3）推理执行。推理的过程就是根据命题和推理的置信度求解其后继命题的置信度的过程。如果最终结论命题的置信度超过规定的阈值则接受该结论，否则拒绝该结论。具体算法如下：

第 1 步，取得第 1 个 *TR* 的测试数据；

第 2 步，运行推理准备过程；

第 3 步，取得第一个变迁；

第 4 步，如果该变迁可以点火则根据其入库所和其自身的置信度计算其所有出库所的置信度，然后标记该变迁为“已点火”；

第 5 步，取得下一个变迁。如果取到，则转第 4 步；

第 6 步，判断本次遍历是否有变迁被点火。如果有则转第 3 步；

第 7 步，取得第一个结论命题库所；

第 8 步，如果该库所的可信度达到阈值，则将其加入推理结果中；

第 9 步，取下一个结论命题库所。如果取到，则转第 8 步；

第 10 步，取下一个 *TR* 的测试数据，如果取到，则转第 2 步；

第 11 步，返回推理结果。

4　实验验证

4.1　实验环境

（1）网络环境。如图 4 所示。

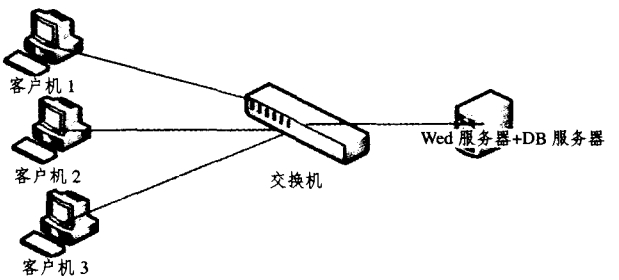

图 4　实验网络环境

（2）计算机软硬件环境。软件配置：客户机-Windows XP Professional，IE6. 0；服务器-Windows 2000 Server，Tomcat 5. 5，Oracle 9i；硬件配置：客户机-CPU P4 2. 0；256M 内存；服务器-CPU P4 2. 4；512M 内存；60G 硬盘；IDE 磁盘控制器。

4.2　实验过程

（1）被测 Web：我们开发的网上书店；

（2）测试脚本：订购图书、查询订单；

（3）测试场景：客户机 1、客户机 2、客户机 3 上均运行 100 个虚拟用户，每个虚拟用户均执行测试脚本。测试过程由客户机 1 控制。

4.3　实验结论

对我们开发的网上书店网站进行了 5 次测试，测试报告显示本文设计和实现的 Web 系统性能优化建议推理方法的有效率达到了 80%。

结论和未来工作　本文采取的基于模糊 Petri 网的推理方法借鉴了人工智能的思想。Web 系统的复杂性和多样性决定了为其建立严格而完备的数学模型是非常困难的。即便可以建立这样的数学模型，其推理前提条件的苛刻也必然影响实际应用。所以，本文另辟蹊径，使用测试专家从实践中获得的经验作为推导的依据，较好的解决了 Web 性能优化建立推理智能化问题。

未来的工作主要在加强推理规则的自学习和推理过程的通用性上。

参 考 文 献

1　中国互联网络信息中心. 中国互联网络发展状况统计报告. http://www. cnnic. net. cn/. 2006

2　Software QA/Test Resource Center. http://www. softwareqatest. com

3　Peterson, Lyle J. Petri net theory and the modeling of system. 1981. 10～13

4　Chen Shyi-Ming , Shiau Yuh-Shin . Vague Reasoning and Knowledge Representation Using *Extended Fuzzy Petri Nets. Journal of* Inforamtion Science and Engineering, 1998, 14(2): 391～408

5　尚涛，牛连强，唐任远. IP 欺骗的技术分析及防御措施. 微计算机信息，2002，18(4)