第44卷第1期 中国科学技术大学学报 Vol. 44, No. 1
2014年1月 JOURNAL OF UNIVERSITY OF SCIENCE AND TECHNOLOGY OF CHINA Jan. 2014

文章编号:0253-2778(2014)01-0012-07

# 基于用户行为模型的TVOS资源分配算法

陈 磊,王 嵩,吴 刚

(中国科学技术大学自动化系网络传播系统与控制安徽省重点实验室,安徽合肥 230027)

**摘要**:现有的智能电视操作系统(TVOS)资源分配多依赖于操作系统本身对任务的资源分配方案,而系统对任务的调度是尽力而为(best-effort)的,以最大化系统的吞吐量为目的,这种资源调度分配方案在实时或多媒体应用系统存在不能保障应用的服务质量(QoS). 为此,在研究TVOS用户行为模型的基础上,量化了用户对应用的偏好,并结合应用QoS模型提出了两种资源分配算法RA_DP和RA_PLSH. 实验结果表明,基于动态规划的RA_DP算法能够求出问题的最优解,可作为算法间比较的参考,但算法时间复杂度很高;基于资源定价的局部搜索启发式RA_PLSH算法可在短时间内求出问题的近似最优解,与其他启发式算法相比更适合于智能电视资源的实时分配.

**关键词**:TVOS;资源分配;用户行为模型;服务质量

**中图分类号**:TP393 **文献标识码**:A doi:10.3969/j.issn.0253-2778.2014.01.002

**引用格式**:Chen Lei, Wang Song, Wu Gang. User behavior model based TVOS resource allocation[J]. Journal of University of Science and Technology of China, 2014,44(1):12-18.
陈磊,王嵩,吴刚. 基于用户行为模型的TVOS资源分配算法[J]. 中国科学技术大学学报,2014,44(1):12-18.

# User behavior model based TVOS resource allocation

CHEN Lei, WANG Song, WU Gang

(*University of Science and Technology of China, Department of Automation, Network Communication System and Control, Hefei* 230027)

**Abstract**: The resource allocation of the existing smart TV operating system (TVOS) depends on the operating system task resource allocation scheme, which tries to maximize the throughput of the system. However, this scheme cannot guarantee the quality of service (QoS) of applications in the real-time or multimedia system. In order to solve this problem, the user preferences of applications based on the user behavior model of TVOS were quantified, and two resource allocation algorithms named RA_DP and RA_PLSH were proposed based on the application QoS model. The experimental result shows that the algorithm based on dynamic programming (RA_DP) ensures optimal solution with high time complexity, and can be used as a reference for other algorithms. However, the other algorithm based on resource pricing local search heuristic (RA_PLSH) can obtain near-optimal solution in a very short time, and is thus more suitable for smart TV real-time resource allocation.

**Key words**: TVOS; resource allocation; user behavior model; QoS

**收稿日期**:2012-10-15;**修回日期**:2013-03-16

**作者简介**:陈磊,男,1986年生,博士生. 研究方向:多媒体资源系统. E-mail: chl123@mail.ustc.edu.cn
**通讯作者**:吴刚,博士/教授. E-mail: wug@ustc.edu.cn

## 0　引言

随着三网融合的演进，新一代的电视终端-智能电视(smart TV)-应运而生. 在 2011 年国际消费电子产品展(CES)上，各大电视厂商推出了基于开放平台的拥有自身操作系统(TV operation system，TVOS)的智能电视产品，昭示了产业发展趋势. 目前已上市的智能电视使用的操作系统大体可分为三类：Android，Windows，企业基于开源 Linux 的自建系统. 这些系统多是从 PC 类(Linux，Windows)或手机类(Android，MeeGo，iOS)移植过来的，若直接在多媒体类型的智能电视终端上应用会存在一些问题. 考虑到嵌入式平台的特点，智能电视面临如下亟待解决的问题：终端的硬件能力虽然不断提高，然而高品质的应用对资源的需求也日益提高，在嵌入式终端上，依然面临着资源受限的问题；操作系统对多任务的支持加剧了任务间对资源的竞争；原有的操作系统尽力而为的调度任务，目的是为了最大化系统吞吐量，不能保障应用的 QoS 和用户体验；各应用之间没有区分度，导致资源分配不合理引起资源浪费. 因此，本文通过研究 TVOS 中的用户行为模型，量化用户对应用的偏好，并结合应用 QoS 模型对系统中运行的多个应用进行资源的优化分配和调度，对保障应用 QoS，提高系统用户体验，达到资源的有效利用.

## 1　相关研究

目前，在资源分配框架、理论模型以及算法方面，许多学者做了大量研究. Rajkumar 等[1]提出了一种 QoS 管理的分析模型 Q-RAM(QoS-based resource allocation model)，在满足应用最小资源需求的条件下达到最大化系统整体效用，并给出了单资源多 QoS 维度下的资源分配算法. 之后 Rajkumar 又研究了多资源单 QoS 维度约束条件下资源分配问题，假设系统效用与资源消耗具有线性关系，建立了混合整数规划模型并求解[2]，但这种假设限制了算法在实际系统中的可用性. Lee 等[3-4]在 Q-RAM 模型基础上，提出了基于离散 QoS 指标的资源分配模型，并分别研究了单资源多 QoS 维度和多资源多 QoS 维度条件下的资源分配问题，给出了基于回溯法的最优算法和基于凸优化理论的近似算法.

Gertphol 等[5]建立了资源分配的混合整数规划模型，但是该方法需要对目标函数进行线性化处理，限制了问题的适用性，同时，算法运行时间随问题规模的增加而指数级增长，因此只适用于离线的资源预分配，不适合在线实时决策. Harada 等[6]通过引入 QoS 自适应控制模块，基于当前任务 QoS 等级和它们 QoS 均值之间的偏差进行反馈，并提出以 QoS 公平管理为目标的自适应资源分配方法.

Khan[7]首次将多会话多媒体系统服务质量自适应问题转换为多维背包问题(MMKP)，以系统整体效用最大作为优化目标，并给出基于分支限界的 BBLP 算法，以求得问题的最优解；同时还提出一种启发式算法 M_HEU，可以在多项式时间内求得近似最优解. 该算法主要结合了 Toyoda[8]提出的用于解决 MDKP 问题的资源消耗总额(aggregate resources consumption)概念来解决 MMKP 问题. Akbar 等[9-10]改进了 M_HEU 算法，提高了求解速度. 伍之昂等[11]也基于资源消耗总额的思想，提出了类似 M_HEU 的启发式算法，可以在多项式时间内求得问题的近似解.

综上所述，现有的资源分配和算法研究，从优化目标上可以分为两类：第一类是以系统的整体效用最大化为优化目标，如文献[1-4]；第二类以系统中各应用的公平性为优化目标，如文献[6]. 第二类问题不适用于 TVOS 的资源分配情况，TVOS 需要尽量保障偏好应用的 QoS 等级，资源分配具有一定的偏向性. 本文重点讨论第一类问题，目前这类问题大多侧重于在满足资源约束前提下，最大化系统整体效用，但在资源分配的过程中没有考虑到各应用之间的区分度，在系统资源比较充裕的情况下，不能更好地提高重要性较高应用的 QoS 水平. 本文从用户的角度出发，建立 TVOS 用户行为模型，量化用户对应用的偏好，以此作为应用间的区分度；然后建立了一种具有应用区分度的 QoS 保障资源分配模型，并提出了两种资源分配算法. 第一种是基于动态规划的 RA_DP(resource allocation based on dynamic programming)算法，能够求出问题的最优解，但时间复杂度随着问题规模增大而指数级的增大，不适合 TVOS 嵌入式终端的实时资源分配；另一种是基于资源定价的局部搜索启发式算法 RA_PLSH(resource allocation based on pricing local search heuristic)，该算法可以在多项式时间内求得问题的近似解.

## 2　用户行为模型

随着 APP-store 上应用资源的丰富，用户可以

在智能电视上安装自己感兴趣的应用.用户在使用智能电视时,除了观看电视视频这个主体服务,同时还可能使用其他应用,如游戏、QQ、网页浏览等.根据兴趣,用户会对某些应用更加"偏好".这部分应用的QoS对用户的体验起决定性的作用.在应用资源分配时,系统需要尽量保证这部分应用的QoS等级.

要建立用户行为模型,就需要构建用户行为信息采集模块,收集用户实际的行为信息.Verkasalo[12]提出了一种智能手机终端的用户行为测量框架MobiTrack,并利用采集的数据进行建模和相关性分析.Gerber等[13]提出了一个基于情景感知的移动手机终端用户模型框架PersonisJ,能够根据用户的行为信息提出个性化的服务,并根据此模型框架开发跨多移动终端的客户端用户模型应用.这部分不是本文的重点,不作详细讨论.

### 2.1 用户偏好度

$[t_{1k},A_j,t_{2k}]$表示应用$A_j$在$t_{1k}$时刻进入终端(实际运行),$t_{2k}$时刻离开终端,则应用此次的运行时间$t_j[k]=t_{2k}-t_{1k}$.用户从进入终端到离开终端,在此过程中进行的一系列操作,称为用户的一次会话.用户的行为信息可以描述为在一个给定时间周期内,用户与系统间发生的一组连续的会话事件序列.为了解用户的行为,需要统计整个会话或长时间周期内用户的行为信息.

用户偏好度是指用户在使用终端时偏好某一应用的概率,体现了用户对应用的偏好程度,同时也能反映各应用相对于系统的重要程度.本文采用时间比率模型来描述.

在给定的时间$[T_s,T_e]$,系统需要记录的用户行为信息:$m_j$表示应用$A_j$运行的总次数;第$k$次运行的时间为$t_j[k]$.令用户对应用$A_j$偏好度为$\mathrm{CON}(A_j)$,若系统的应用总数量为N,则用户在这段时间内对应用$A_j$的偏好度可以表示为:

$$\mathrm{CON}(A_j)=D_j/\sum_{q=1}^{N}D_q,\ D_j=\sum_{k=1}^{k=m_j}t_j[k] \quad (1)$$

式中,$D_j$表示给定时间内$A_j$运行的总时间.由式(1)可知,$\mathrm{CON}(A_j)$是位于$[0,1]$区间内的实数,数值越大,表示该应用用户更偏好,对系统而言更关键.

### 2.2 模型更新

在资源分配过程中,需要的是当前时刻的用户偏好度.通过统计用户在过去时间内所有的行为信息,以后验概率作为当前时刻的用户偏好度估计.另外,用户的偏好度随着用户习惯和兴趣的转移会发生变化,若采用累积的行为统计会导致无法即时反馈用户的近期偏好.针对这个问题,引入时间衰减函数对用户偏好度进行更新.

假设用户偏好度在一天内是稳定的,记录用户每天对应用的$A_j$行为信息,若已收集了L天的用户行为信息,那么当前时刻用户对应用$A_j$的偏好度:

$$\mathrm{CON}(A_j,T_c,0)=\sum_{p=1}^{L}(\mathrm{CON}(A_j,T_c,p)\times f(p))/\sum_{p=1}^{L}f(p) \quad (2)$$

式中,$\mathrm{CON}(A_j,T_c,p)$表示距离当前时刻$p$应用$A_j$的实测偏好度,利用式(1)计算可得.$f(p)=e^{-\lambda p}$,$(\lambda>0)$为引入的指数时间衰减函数.

## 3 保障应用QoS的资源分配算法

### 3.1 应用、资源和QoS维度

考虑TVOS中有$n$个应用$\{A_1,A_2,\cdots,A_n\}$同时运行,并具有$m$种不同的资源$\{R_1,R_2,\cdots,R_m\}$.每种资源的上限表示为$r^{\max}=\{r_1^{\max},r_2^{\max},\cdots,r_m^{\max}\}$,在时间上或空间上由$n$个任务共享.例如,CPU和网络带宽是时间共享资源,而内存和磁盘是空间共享资源.

应用为用户提供服务需要消耗系统资源,应用的QoS取决于系统分配的资源.应用获得的资源越多,其QoS等级越高,应用的性能也越好.应用可根据所分配的资源水平方案工作在不同的QoS等级上.应用的QoS需要由系统或应用的开发者事先定义.例如,考虑视频会议应用,其中涉及实时的音视频流的加密和在不可靠的网络中的数据流的传输,QoS各维度可定义为:图像格式、帧率、音频采样率、色深、端到端时延等.

令$r_{ij}$表示分配给应用$A_i$的资源$j$的数量,则应满足:$\sum_{i=1}^{n}r_{ij}\leqslant r_j^{\max}$.应用的性能可以通过其效用值$U_i$来衡量[1].令$\overrightarrow{r_{ik}}=(r_{i1k},r_{i2k},\cdots,r_{imk})$,表示应用$A_i$运行在资源分配方案$k$时所具备的资源分配量,应用在此时获得的效用值为$u_i(\overrightarrow{r_{ik}})$.当$\overrightarrow{r_{i0}}=(0,0,\cdots,0)$时,此方案不为应用分配资源,应用也就无法运行,效用$u_i(\overrightarrow{r_{i0}})=0$.文献[2,4]中分别用非递减的线性凹函数和离散工作点集合来表示R-U关系.第一种表示在实际系统中过于理想化,一般的研究采用第二种表示.

### 3.2 应用QoS模型描述

本文做如下假设:

① 系统中各应用之间是独立的；

② 系统的可用资源能满足每个应用运行的最小资源需求；

③ 用户对应用 $A_i$ 的偏好作为权重系数 $w_i=CON(A_i,T_c,0)$，$w_i$ 区分各应用对于系统的重要性；

④ 应用 $A_i$ 具有 $N_i$ 种不同的资源分配方案，提供给用户 $N_i$ 不同的应用 QoS 等级，同时对应 $N_i$ 种不同的性能质量水平，获得的资源越多，应用的性能越好.

对整个系统而言，同时运行着多个应用，每个应用对应着一种资源分配方案. 定义系统整体效用值为所有应用所获得的效用的加权和：

$$u_{total}=\sum_{i=1}^{n}w_iu_i \tag{3}$$

系统资源分配的目标是为 $n$ 个应用合理地分配资源，在满足系统资源约束的条件下，通过确定各应用资源的分配方案，最大化系统的整体效用，即

$$\left.\begin{aligned}&\text{Max: } u_{total}=\sum_{i=1}^{n}\sum_{k=1}^{N_i}\chi_{ik}\cdot w_iu_i(r_{ik})\\&\text{st:}\\&R_{cj}=\sum_{i=1}^{n}\sum_{k=1}^{N_i}\chi_{ik}\cdot r_{ijk}\leqslant r_j^{max},\ j=1,2,\cdots,m\\&\sum_{k=1}^{N_i}\chi_{ik}=1,\ \chi_{ik}\in\{0,1\},\\&\qquad i=1,2,\cdots,n;\ k=1,2,\cdots,N_i\end{aligned}\right\} \tag{4}$$

式中，$\chi_{ik}$ 是示性函数，表示在应用 $A_i$ 在运行时，只能从 $N_i$ 种资源分配方案选择一种资源分配方案. 这种整体效用值定义方式，相比于 $u_{total}=\sum_{i=1}^{n}u_i$，为了最大化系统整体效用，偏好度较高的应用在资源分配时有机会获得更多的资源，表现出更高的 QoS 等级.

该资源分配模型可以简化为一个 0～1 背包问题，文献[4]证明其是一个 NP 难问题. 下文分别给出了两种资源分配算法. 为了方便算法的阐述，假设系统的资源数为 $m=2$，应用 $A_i$ 的 R-U 离散工作点集合表示：

$$C_i=\left[\begin{pmatrix}u_{i1}\\\overrightarrow{r_{i1}}\end{pmatrix},\cdots,\begin{pmatrix}u_{iN_i}\\\overrightarrow{r_{iN_i}}\end{pmatrix}\right],$$

以效用值大小非递减排列. 其中，$\overrightarrow{r_{ik}}=(r_{i1k},r_{i2k})$ 代表应用 $A_i$ 可运行的一种资源分配方案，存在关系式：

$$C_i(\overrightarrow{r_{ik}})=u_{ik}\quad 或\quad C_i(r_{i1k},r_{i2k})=u_{ik}.$$

### 3.3　RA_DP 算法

RA_DP 算法是基于动态规划的资源分配算法，算法的主要思想是找出资源分配问题的最优子结构，递归的定义最优解，以自底向上的方式求出最优解. 假设资源的分配是以资源单元来进行的，若存在 $r_i^{max}/S_i$，当 $S_i=100$ 时，资源是以百分比单元的整数倍分配的.

这里以二维资源情况（$m=2$）为例来阐述算法，最优资源分配的问题描述如下：

令 $v(i,s_1,s_2)$ 表示当第 $i$ 个应用的资源分配后系统获得的最大效用值，$s_1$，$s_2$ 表示可用的资源量，$w_i$ 表示应用的偏好权重，则问题的递归最优子结构表示为：

$$v(i,s_1,s_2)=\max_{\substack{s_1'\in\{0,\cdots,s_1\}\\s_2'\in\{0,\cdots,s_2\}}}\{w_iC_i(s_1',s_2')+v(i-1,s_1-s_1',s_2-s_2')\} \tag{5}$$

式中，$s_1'$，$s_2'$ 是工作点集 $C_i$ 中实际的离散工作点. $v(n,S_1,S_2)$ 则表示当 $n$ 个应用资源都分配完后系统获得总的最大效用值. 算法在递推过程中，为了能最大化每个阶段的累积效用值，更倾向于将资源分配给 $w_i$ 更高的应用. RA_DP 算法执行过程如图 1 所示.

令 $L=\max_{i=1}^{n}|C_i|$，则 RA_DP 算法的时间复杂度为 $O(nLS_1S_2)$ 或 $O(nS_1^2S_2^2)$，扩展到更多维资源的情况，时间复杂度变为 $O(nS_1^2\cdots S_m^2)$ 或 $O(nS_x^{2m})$. 由此可见，随着问题规模的增大，算法的时间复杂度呈指数级增大，所以这种资源分配方案虽然能够得到问题的最优解，但是时间复杂度太高，不适合用于 TVOS 的实时决策，只可作为离线资源的预分配或作为其他算法的性能参考.

### 3.4　RA_PLSH 算法

由于 RA_DP 算法的时间复杂度过高，本节提出一种基于资源定价的局部搜索启发式算法 RA_PLSH，可以在多项式时间内求得问题的近似最优解. 本算法主要思想是优先选择单位资源下能够提供较大效用值的资源分配方案，在各应用工作点集的凸包[14]边界上局部搜索工作点，从而获得较大的系统整体效用值.

为了解决多资源问题导致问题规模增大的问题，通过引入复合资源将多维资源向量转换为单维资源标量. 例如，某个资源分配方案 $\overrightarrow{r}=(r_1,\cdots,r_m)$，其复合资源向量可表示为：

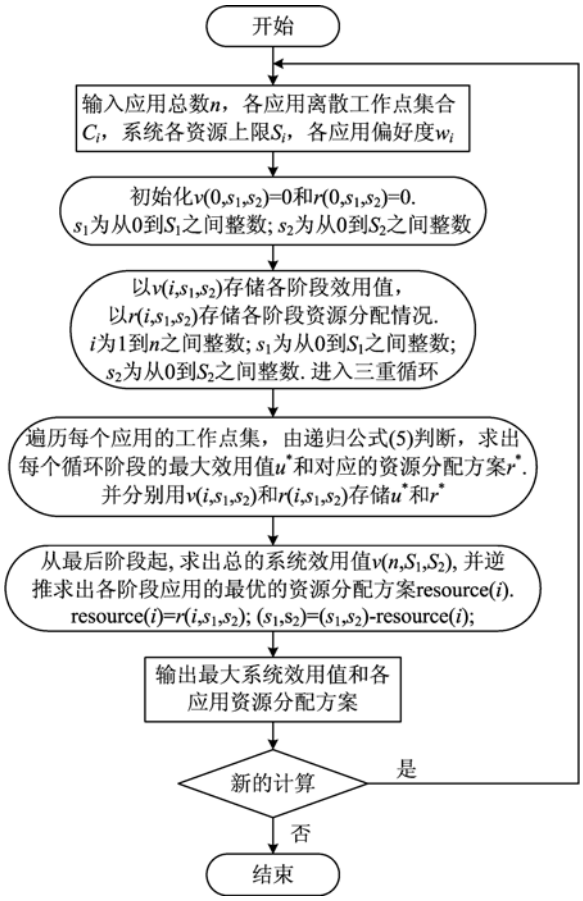


**图 1 RA_DP 算法流程图**

**Fig. 1 The flow chart of RA_DP algorithm**

$$r^* = \| \vec{r} \| = \sqrt{(r_1)^2 + \cdots + (r_m)^2}.$$

将 $r^*$ 加入到应用工作点集并考虑各应用的偏好权重，将应用 $A_i$ 的离散工作点集合更新为：

$$C_{ic} = \left[ \begin{pmatrix} u^*_{i1} \\ \rightarrow \\ r_{i1} \\ r^*_{i1} \end{pmatrix}, \cdots, \begin{pmatrix} u^*_{iN_i} \\ \rightarrow \\ r_{iN_i} \\ r^*_{iN_i} \end{pmatrix} \right],$$

其中，$u^*_{ik} = w_i \cdot u_{ik}$.

为了介绍算法，首先介绍资源定价[8]的概念，它表示应用从一个资源分配方案 $p$ 到另一个方案 $q$，单位资源所产生的效用值的变化：

$$\Delta u_{i,(p \to q)} = (C_{ic,q} \cdot u^* - C_{ic,p} \cdot u^*)/(C_{ic,q} \cdot r^* - C_{ic,p} \cdot r^*) \quad (6)$$

选取 $C_{ic}$ 中的复合资源和效用值分别作为二维坐标系下的横坐标和纵坐标，则各应用对应的离散工作点分布在二维坐标的第一象限内. 为了优先选取能产生最大资源定价的工作点，对应用二维坐标系中的离散工作点构建凸包，在凸包的边界(连接最低点到最高点的各个分段)点上进行局部搜索.

RA_PLSH 的算法运行过程如下：

输入：各应用离散工作点集合 $C_i$，系统各资源上限 $S_i$，各应用偏好度 $w_i$

输出：各应用选择的资源分配方案索引号构成的解方案 X，系统效用值 U

(Ⅰ) 以各应用最小的资源分配方案索引号作为初始化解方案 X=(1,1,…,1)；

(Ⅱ) 保存当前解方案 $X^* \leftarrow X$，求出当前解方案对应的系统效用值 $U^* \leftarrow$Utility($X^*$)；

(Ⅲ) 遍历各应用的离散工作点集合 $C_i$，依次完成以下过程：

(*a*) 将 $C_i$ 转换为复合资源代表的离散工作点集合 $C_{ic}$；

(*b*) 构建二维坐标系下 $C_{ic}$ 的凸包，凸包边界呈现非递增的斜率分段，以方便算法每次选择的都是当前剩余方案中具有最大资源定价的方案；

(*c*) 将凸包边界上的分段加入到全局 frontier_segments 集合中；

(Ⅳ) 将 frontier_segments 集合中所有分段端点按照斜率非递增的顺序排列为 v_list；

(Ⅴ) 遍历 v_list 中的点对应的资源分配方案，依次验证是否满足资源约束条件，若满足，则调整当前解方案中应用对应的资源分配方案索引号，更新解方案和系统剩余资源；若不满足，则不更新；

(Ⅵ) 执行完(Ⅴ)后得到新的解方案 X，计算此时系统效用值效用值 U，若 $U < U^*$，则恢复保存解，$X \leftarrow X^*$；

(Ⅶ) 算法结束后，返回获得的最大系统效用值 U 和各应用的资源分配解方案 X.

该算法时间复杂度为 $O(nLm + nL\log nL)$，通过引入复合资源向量解决了由于资源数的增加带来的时间开销，该算法可以在多项式时间内解决系统的资源分配问题，适合 TVOS 的实时决策.

## 4 仿真实验

本节通过仿真实验分析算法的性能. 因为 RA_DP 算法能求出资源分配问题的最优解，在性能比较时将其作为参照标准；RA_PLSH 是启发式的快速求解问题的算法，本节将其与另一种流行的启发式算法 M_HEU[7,11] 进行性能对比，M_HEU 算法的时间复杂度为 $O(mn^2L^2)$.

算法的性能指标的主要包括算法的执行时间、

求得的系统整体效用值. 实验以 RA_DP 算法的性能值作为参照标准,定义如下比较指标:

执行时间比,即 RA_DP 算法的执行时间分别与 RA_PLSH 算法和 M_HEU 算法的执行时间之比:

$$\gamma_1 = t_{RA\_DP}/t_{RA\_PLSH},\ \gamma_2 = t_{RA\_DP}/t_{M\_HEU}.$$

系统整体效用值比,即 RA_PLSH 算法和 M_HEU算法求出的近似解分别于 RA_DP 算法获得的最优值之比:

$$\beta_1 = U_{RA\_PLSH}/U_{RA\_DP},\ \beta_2 = U_{M\_HEU}/U_{RA\_DP}.$$

利用 Matlab 程序实现对测试方案的编写,测试方案分为两组,第一组固定资源数 $m=8$,以应用数 $n=\{5,10,15,20,25,30\}$ 为测试案例;第二组固定应用数 $n=20$,以资源数 $m=\{2,4,6,8,10,12\}$ 为测试案列. 针对每组情况,分别随机产生各应用的离散工作点集合和偏好度. 偏好度值在(0,1)区间内.

图 2 给出了随着应用数量增大的实验情况,图 2(a)显示了两种执行时间比曲线的变化情况. 两种启发式的算法都比 RA_DP 算法快三个数量级,其中 RA_PLSH 算法表现更为优异,当 $n=25$ 时,RA_PLSH 算法的执行时间比 M_HEU 算法快两倍左右. 这因为,RA_PLSH 算法通过构建工作点集的凸包,大大减少了算法在运行时遍历的工作点数目;而 M_HEU 虽然也是一种启发式的算法,但是它需要遍历绝大部分的工作点来寻求近似解. 图 2(b)显示了两种算法获得的系统整体效用值比的情况. 当问题规模较小时,两种算法都能获得接近于 100% 的最优值,问题规模增大后,效用值比略有降低,但都维持在 94% 以上. 其中 M_HEU 算法在这方面比 RA_PLSH 算法更能维持高效用比.

图 3 给出了随着资源数量增大的实验情况,图 3(a)显示了两种执行时间比曲线的变化情况. 由图

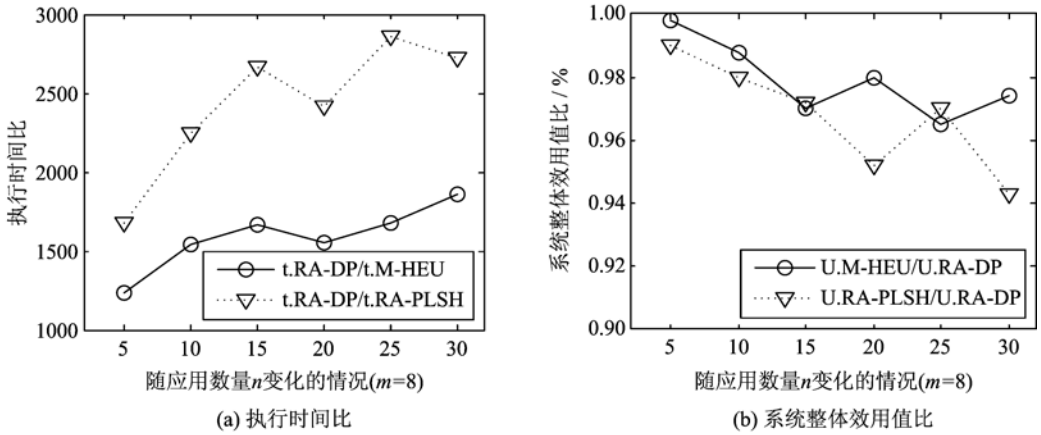


**图 2　第一组实验**

**Fig. 2　The first group of experiments**

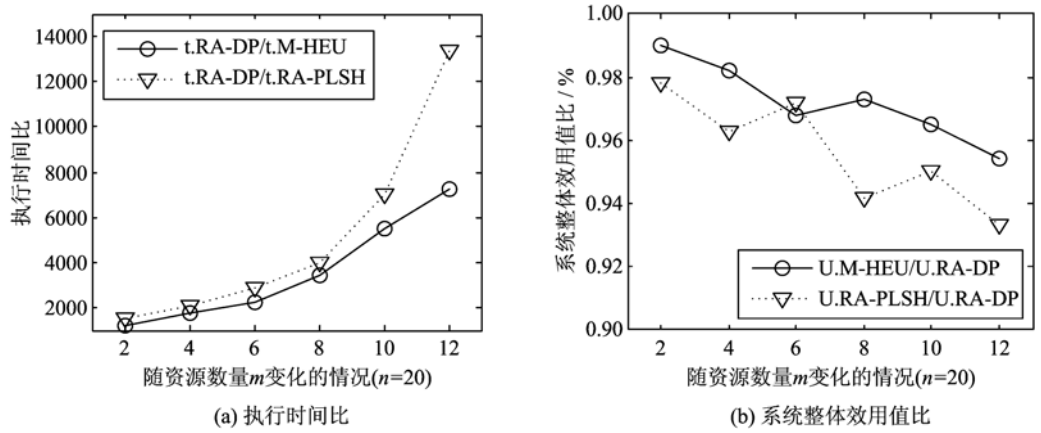


**图 3　第二组实验**

**Fig. 3　The second group of experiments**

3 可见，RA_DP 算法随着问题规模的增大，与另外两种启发式算法的执行时间比成指数级的上升趋势，这在与 RA_PLSH 算法的执行时间比曲线上表现得更为明显. 这是因为其时间复杂度为 $O(nS_x^{2m})$，当资源数增大时，算法复杂度指数级的增大，而 RA_PLSH 算法采用了复合资源将多维资源向量转换为单维资源标量，避免多资源带来的算法扩展问题. 图 3(b)显示了两种算法获得的系统整体效用值比的情况. 两种算法都能维持 92%以上的效用值比.

通过以上两组实验，RA_PLSH 算法在解决大规模问题时，与 M_HEU 算法相比具有更高的执行速度优势，虽然可能会损失些近似最优解，但考虑到对实时性要求甚高的 TVOS，显然算法的求解速度应该是首位的，并且 RA_PLSH 算法同样能维持 92%以上的效用值比，在可接受的范围内，所以，RA_PLSH 算法比 M_HEU 算法更适合于实时系统 TVOS 的资源分配. RA_DP 算法能虽然求解出最大的系统效用值，但付出的时间代价太大，不适于 TVOS 的实时处理，可作为其他算法的性能参考.

## 5 结论

本文主要研究 TVOS 中的多资源分配问题. 首先从用户角度出发，建立 TVOS 用户行为模型，量化用户对应用的偏好，在此基础上，结合应用 QoS 模型，提出了两种具有应用 QoS 保障的资源分配算法. 一种是基于动态规划的思想的算法 RA_DP，另一种是基于资源定价的局部搜索启发式算法 RA_PLSH. 实验结果表明，RA_DP 可以求出问题的最优解，但算法的时间复杂度是随着问题规模增加呈指数级增大，不适合嵌入式终端的实时决策，可以用于算法间的参考对比；RA_PLSH 可以在短时间内求出问题的近似解，与最优解相比只损失了 8%，与其他启发式算法相比，更适合于 TVOS 资源分配的实时处理.

**参考文献(References)**


[1] Rajkumar R, Lee C, Lehoczky J, et al. A resource allocation model for QoS management [C]// Proceedings of the 18th IEEE Real-Time Systems Symposium. San Francisco, USA: IEEE Press, 1997: 298-307.

[2] Rajkumar R, Lee C, Lehoczky J P, et al. Practical solutions for QoS-based resource allocation problems [C]// Proceedings of the 19th IEEE Real-Time Systems Symposium. Madrid, Spain: IEEE Computer Society, 1998: 296-306.

[3] Lee C, Lehoczky J, Rajkumar R, et al. On quality of service optimization with discrete QoS options[C]// Proceedings of the 5th IEEE Real-Time Technology and Applications Symposium. Vancouver, Canada: IEEE Computer Society, 1999: 276-286.

[4] Lee C, Lehoczky J, Siewiorek D, et al. A scalable solution to the multi-resource QoS problem [C]// Proceedings of the 20th IEEE Real-Time Systems Symposium. Phoenix, USA: IEEE Computer Society, 1999: 315-326.

[5] Gertphol S, Prasanna V K. MIP formulation for robust resource allocation in dynamic real-time systems [J]. Journal of Systems and Software, 2005, 77(1): 55-65.

[6] Harada F, Ushio T, Nakamoto Y. Adaptive resource allocation control for fair QoS management[J]. IEEE Transactions on Computers, 2007, 56(3): 344-357.

[7] Khan S. Quality adaptation in a multisession multimedia system: Model, algorithms and architecture[D]. Department of ECE, University of Victoria, 1998.

[8] Toyoda Y. A simplified algorithm for obtaining approximate solution to zero-one programming problems [J]. Management Science, 1975, 21(12): 1 417-1 427.

[9] Akbar M M, Manning E G, Shoja G C, et al. Heuristic solutions for the multiple-choice multi-dimension knapsack problem[C]// Proceedings of the International Conference on Computational Science. San Francisco: Springer-Verlag, 2001: 659-668.

[10] Shahriar M, Akbar M M, Rahman M S, et al. A multiprocessor based heuristic for multi-dimensional multiple-choice knapsack problem[J]. The Journal of Supercomputing, 2008, 43(3): 257-280.

[11] 伍之昂，罗君舟，宋爱波，等. 具有 QoS 保证的服务资源联合分配与管理[J]. 软件学报，2009，20(12)：3 150-3 162.

[12] Verkasalo H. Analysis of smartphone user behavior [C]// Proceedings of the 9th International Conference on Mobile Busines/9th Global Mobility Roundtable. Athens, Greece: IEEE Computer Society, 2010: 258-263.

[13] Gerber S, Fry M, Kay J, et al. PersonisJ: Mobile, client-side user modelling[C]// Proceedings of the18th International Conference on User Modeling, Adaptation, and Personalization. Big Island, USA: Springer, 2010: 6075: 111-122.

[14] 维基百科. 凸包[EB/OL]. http://zh.wikipedia.org/wiki/%E5%87%B8%E5%8C%85.