基于时间约束网络的智能活动规划

李永峰[1] 周兴社[1] 杜可君[1] 於志文[1] 毛 睿[2]
(西北工业大学计算机学院 西安 710129)[1] (中国移动(深圳)有限公司 研发支撑部 深圳 518034)[2]

摘 要 老年人认知能力的下降导致其无法正常规划日常生活的问题已经越来越受到社会的关注。利用信息技术辅助老年人独立完成日常活动,已成为目前一个新的研究领域,其中对其活动的规划和提醒是该领域的一个研究热点。在传统基于时间的活动约束表示和冲突检测的基础上,提出一种更为宽松合理的时间约束,其使活动时间更为灵活,同时引入活动规划中新的活动时间冲突问题。通过时间约束网络的相关概念和理论,将活动规划中冲突检测问题转化为简单时间约束网络是否满足一致性的问题。给出时间约束一致性的一般性检测算法,分析得出活动数较大时其算法复杂度呈指数增长。针对一般性检测算法,提出新的检测时间约束一致性的算法,以降低算法复杂度,解决活动规划中的时间冲突问题。最后通过实验对两种算法的时间复杂度进行了比较。

关键词 活动规划,时间约束,简单时间约束网络,一致性
中图法分类号 TP399 **文献标识码** A

Activity Planning Based on Temporal Constraint Network

LI Yong-feng[1] ZHOU Xing-she[1] DU Ke-jun[1] YU Zhi-wen[1] MAO Rui[2]
(Institute of Computer Science and Technology, Northwestern Polytechnical University, Xi'an 710129, China)[1]
(Department of Research and Support, China Mobile Limited, Shenzhen 518034, China)[2]

Abstract Elders often have difficulties in daily activities because of declining cognitive ability. Assisting elders with information technology has become a new and challenging research area. It's a problem that how to pre-defining activities into a plan and prompting the elders if they fail to execute it in this area. We gave a more flexible way to present the time constraints of activities based on the traditional way, but as the same time how to resolve the conflicts between activities become a new troublesome problem. The concept and theory of "Simple Temporal Constraint Network"(STCN) was then proposed, which will well handle the problem. We presented the time constraints of activities with STCN and checked the STCN's consistency. The "General Checking Algorithm"(GCA) was adopt, but we found that the GCA's complexity will increase by exponentially if the number of activities increases. Then, we gave an improved algorithm to resolve the conflicts between activities and compared the two algorithms. The experiment result shows the ICA has better performance than GCA.

Keywords Activity planning, Temporal constraint, Simple temporal constraint network, Consistency

随着年龄的增长而导致记忆力日渐衰退,是大多数老年人面临的问题。COGKNOW[1]项目中通过对瑞典的吕勒奥、荷兰的阿姆斯特丹和英国的贝尔法斯特3处老年人生活辅助实验中心的多位用户进行调研发现,高达85%的老年人对自己的记忆能力表示担忧。针对调查表中的23项日常活动,仅有15%的老年人对自己的记忆能力表示自信,并且不需要任何提醒帮助;其余85%的老年人则对自己的记忆力表示担忧,其中更有60%以上的老年人表示会经常忘事,需要他们的生活照顾者协助安排日常活动,并借助提醒设备进行辅助。因此,合理地安排老年人的活动,通过提醒服务来辅助活动的执行,是老年人活动辅助的重要内容。

从活动辅助的阶段来看,安排老年人的活动主要是离线(off-line)阶段的活动规划问题,而活动提醒更多的是在在线(on-line)阶段辅助活动的执行。本文重点讨论活动规划的问题。

活动规划问题简单来说,就是对老年人在什么时间完成什么活动做计划。计算机系统只有很清楚地了解老年人的活动计划,才可能对其活动进行相关的辅助。传统的老年人活动规划大多是单一的基于时间的模式,只考虑每个活动的起止时间,然后形成一张时间表,作为活动计划。在这种简单的时间表规划方式的基础上,许多研究者试图加入更多的约束因素,以求制定更加完备的活动计划,其中ROBOCARE[2,3]

到稿日期:2010-03-30 返修日期:2010-08-09 本文受国家自然科学基金(60903125,60803044),国家863高技术研究发展计划基金项目(2009AA011903),教育部"新世纪优秀人才支持计划",教育部博士点基金新教师课题(20070699014)资助。

李永峰(1985-),男,硕士生,主要研究方向为普适计算;周兴社(1955-),男,博士,教授,博士生导师,CCF常务理事,主要研究方向为嵌入式计算、分布式计算、普适计算;杜可君(1981-),男,博士生,主要研究方向为普适计算;於志文(1977-),男,博士,教授,主要研究方向为普适计算;毛 睿(1979-),男,主要研究方向为通信工程。

和 Autominder[4-6] 是目前比较有代表性的系统。ROBO-CARE 在活动起止时间的基础上加入了活动间的相互约束关系来完善活动规划。Autominder 将老年人的活动规划问题描述成一个析取时态问题(disjunctive temporal problems, DTPs)[7,8],将每个活动以及活动之间的时间约束关系用形式化的方式进行描述,并判断活动计划是否可执行。本文在传统的时间模式的基础上加以改进,提出一种更加灵活的时间约束。为老年人规划的活动包括最早开始时间(T_{ES})、活动的截止时间(T_D)和活动的持续时间(Dur),在活动的 T_{ES} 和 T_D 之间老年人可以完成该活动,且活动的持续时间应大致和 Dur 一致。

从系统输入的角度来看,可以认为目前老年人活动规划系统的输入主要是时间约束。根据我们的用户调研发现,活动计划的制定者主要包括医生、家庭护士、子女以及老年人自己 4 类。角色的不同会使他们给老年人制定不同的活动任务,并且各自的活动计划往往是在不同地点、不同时间分别制定的,因此客观上他们无法在活动的时间上相互协调,从而导致制定的活动在时间上相互冲突甚至无法完成。对于本文提出的描述时间约束的方法,要判断所有活动的时间约束条件是否都能得到满足而不存在冲突,是比较困难的。为此我们引入时间约束网络的模型来解决此问题。

时间约束网络的概念来自于人工智能领域,最早由 Dechter[9] 提出。其主要思想是将时间约束问题描述成图论中的网络约束问题,通过图论中处理网络约束问题的方法来寻求时间约束问题的解。本文将新定义的时间约束转化为简单时间约束网络,通过检测简单时间约束网络的一致性解决了上述提出的各活动时间约束条件是否能得到满足的问题。

本文首先介绍了在老年人活动规划问题中传统时间约束的表示,提出了一种更为宽松的时间约束模式,并引入时间约束网络的概念来求解活动的时间约束条件是否得以满足。本文第 2 节给出了传统的时间约束并加以改进;第 3,4 节详细介绍了时间约束网络的概念和理论,将时间约束问题转化为时间约束网络问题;第 5 节详细分析了时间约束网络一致性检测的一般性算法和改进算法的复杂度;最后总结全文。

1 传统时间约束及改进

合理地安排老年人在不同的时间从事不同的活动,是基于时间约束的活动规划的本质。医生、家庭护士、子女以及老年人自己根据各自关心的部分制定不同的活动,并输入活动的时间约束。智能规划系统将这些时间约束作为输入,判断各活动之间是否存在冲突。如果没有冲突,则将所有活动组成的时间表作为活动的执行计划;否则将冲突信息反馈给活动的制定者,告知需要修改活动的时间约束。

对智能规划系统来说,根据所有活动的时间约束来判断各活动之间是否存在时间冲突,是其核心内容。最传统的冲突检测方式是判断两个活动的时间范围是否存在交叠。设活动的开始时间和结束时间分别为 T_S 和 T_E,给定活动 A_1 和 A_2,下式为判定两者发生重叠的条件:

$$T_{S_2}\leqslant T_{S_1}\leqslant T_{E_2}\,||\,T_{S_2}\leqslant T_{E_1}\leqslant T_{E_2}\,||\,T_{S_1}\leqslant T_{S_2}\leqslant T_{E_1}\,||\,T_{S_1}\leqslant T_{E_2}\leqslant T_{E_1} \tag{1}$$

图 1 示出了活动 A_1 与 A_2 在时间上发生冲突的 4 种可能情况。

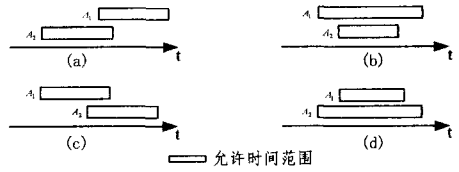


图 1 传统活动时间约束示意图

基于传统时间约束的活动规划方式虽然简单、直观,但其缺点是对活动时间的描述不够灵活。实际上老年人由于活动能力的下降,活动的计划应该设定较为宽松的时间约束。

本文通过三元组 $TC=(T_{ES},T_D,Dur)$ 来表示活动的时间约束,其中 T_{ES}(Earliest Start Time)表示活动的最早开始时间;T_D(Deadline)表示活动的截止时间;Dur 表示活动的最短持续时间,可以由用户直接手动输入,也可以通过长期的统计学习得到。例如吃早餐的时间约束可以描述为:

(7:00am,9:00am,30min)

说明计划中早餐的时间必须在早晨 7 点到 9 点之间完成,早餐一般需要 30 分钟以上。

这种三元组的时间约束描述方式给活动规划带来了更好的灵活性。如图 2 所示,虽然活动 A_1 与 A_2 的允许时间范围发生重叠,但系统认为 A_1 和 A_2 并不冲突,依然可以各自独立得到执行。

图 2 本文定义活动时间约束图

对于采用三元组 $TC=(T_{ES},T_D,Dur)$ 作为描述活动时间约束的方法,要判断所有活动的时间约束条件是否都能得到满足而不存在冲突,是比较困难的。本文引入时间约束网络模型,利用时间约束网络的相关理论来解决老年人活动的时间规划问题。

2 时间约束网络的相关概念及理论

时间约束网络是对具有时间知识和时间约束的系统进行描述和推理的有效方式[10],在任务规划和调度领域得到了广泛的应用。下面首先介绍时间约束网络的相关概念和理论,然后阐述如何利用时间约束网络对活动进行规划。

定义 1(时间约束问题) 时间约束问题定义为一组变量集合 $X=\{X_1,X_2,\cdots,X_n\}$ 和一组变量上的约束集合 $C=\{C_1,C_2,\cdots,C_n\}$。变量集合中的每个元素代表一个时间点,而约束集合中的每个元素表示时间点之间的时间约束关系。

当赋值 $\{X_1=a_1,X_2=a_2,\cdots,X_n=a_n\}$ 满足所有的约束条件时,则称元祖 $X=\{a_1,a_2,\cdots,a_n\}$ 为时间约束问题的解。

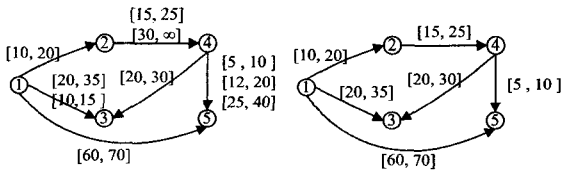
定义 2(时间约束网络) 时间约束网络是时间约束问题的图论描述方式,每个时间约束网络表示为一个有向约束图 $G=\langle V,E\rangle$,其中顶点集合 V 对应时间点集合 X,边集合 E 对应时间约束集合 C。

从 E 中任取一条边,该边对应的两个顶点 $X_i,X_j\in V$ 满足约束条件:

$$(l_1\leqslant X_i-X_j\leqslant u_1)\cup\cdots\cup(l_k\leqslant X_i-X_j\leqslant u_k) \tag{2}$$

也就是说,从 X_j 到 X_i 的时间必须落在时间段 $[l_1,u_1]$,$[l_2,u_2]$,$\cdots$,$[l_n,u_n]$ 之一的范围内。图 3 为时间约束网络的

示意图。



(a)一般时间约束网络　　(b)简单时间约束网络

图 3　时间约束网络示意图

定义 3(简单时间约束网络)　若有向约束图 $G=\langle V,E\rangle$ 每条边的约束条件都为单一约束条件，则称 $G=\langle V,E\rangle$ 描述的时间约束网络为简单时间约束网络。简单时间约束网络对应描述的问题为简单时间约束问题。

图 3(b)为简单时间约束网络的有向图，每条边仅有一个约束条件。例如节点 1 和节点 2，其时间约束条件为：

$$X_2-X_1\in[10,20] \tag{3}$$

该约束条件可以分别表示为两个不等式：

$$\begin{aligned} &X_1-X_2\leqslant-10\\ &X_2-X_1\leqslant20 \end{aligned} \tag{4}$$

将图 3(b)中每一条边的约束条件都表示成式(4)的方式，则简单时间网络可转化为有向加权图 $G_d=\langle V_d,E_d\rangle$ (Dechter 称之为距离图，本文采用有向加权图的说法)。图 4 为图 3(b)转化而来的有向加权图，图中每条边的权值为一具体数值，而不再是如图 3(b)所示的时间区间。

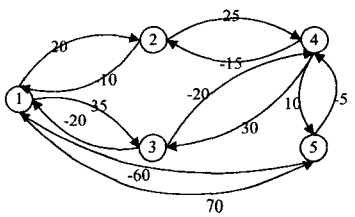


图 4　简单时间约束网络的有向加权图表示

定义 4(最短路径)　设 p 表示节点 i 到节点 j 的一条路径，经过 $i_0=i,\ i_1,\cdots,i_k=j$ 共 $k+1$ 个节点，则路径 p 满足约束：

$$X_j-X_i\leqslant\sum_{j=1}^{k}w_{i_{j-1}i_j}=w_p \tag{5}$$

记 w_p 为路径 p 的权重。若从节点 i 到节点 j 共有 m 条路径，则记 i 到 j 的最短路径为：

$$d_{ij}=\min\{w_{p1},w_{p2},\cdots,w_{pm}\} \tag{6}$$

定义 5(负环)　设 ζ 为有向加权图 $G_d=\langle V_d,E_d\rangle$ 的环，若 ζ 中所有边的权值之和为负数，则称 ζ 为负环。

定义 6(时间约束网络的一致性)　当给定时间约束问题有解时，其所对应的时间约束网络是一致的。

换句话说，当且仅当时间约束网络满足一致性，其所描述的时间约束问题才有解。

定理 1　给定简单时间约束网络是一致的，当且仅当它的有向加权图不存在负环。

3　基于简单时间约束网络的活动描述

由于本文针对的对象主要是记忆力衰退、活动能力有限、日常生活需要照顾的老年人，因此对老年人的活动本文做如下假设：

(1)不考虑单个活动可以安排在多个时间段内的情况，每个活动只能安排在一个固定的时间段内，即满足简单时间约束的特性。

(2)由于老年人的活动能力下降，故不考虑老年人同时并发执行多个活动的情况。也就是说，所有活动的时间约束必须满足简单时间约束的一致性，否则智能规划系统认为活动间有冲突。

活动的时间约束可以从描述信息中得到，并转化为时间约束网络需要的不等式形式，例如(7:00am，9:00am，30min)可以通过图 5 所示的 3 个不等式来等价描述。

早餐最早 7:00am 开始	$T_S-T_R\geqslant T_{ES}=420$
早餐需要 30 分钟以上的时间完成	$T_E-T_S\geqslant Dur=30$
早餐最晚 9:00am 必须结束	$T_E-T_R\leqslant T_D=540$

图 5　时间约束的不等式描述

其中，T_R 为参照时间，本文选用零点作为参照时间，T_S 为活动的实际开始时间，T_E 为活动的实际结束时间，时间的计算单位为分钟。根据简单时间约束网络的定义(定义 3)，图 5 的不等式描述可以转化为图 6 所示的 3 个顶点的有向加权图，所对应边的权重分别为：

$$W_{T_RT_E}=540,W_{T_ET_S}=-30,W_{T_ST_R}=540$$

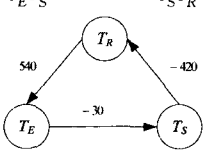


图 6　早餐的简单时间约束网络图

4　时间约束一致性的检测算法

4.1　一般性算法

根据定理 1，可以利用检测是否存在负环来判断活动之间是否满足时间约束。我们首先以两个活动为例来说明。对任意给两个活动 A_1 和 A_2，系统做如下假设：若 $T_{ES1}<T_{ES2}$，则系统认为活动 A_1 应先于活动 A_2 执行，则 A_1 和 A_2 除了各自的时间约束外，还应满足相对约束 $T_{E1}-T_{S2}\leqslant0$。图 7 中粗线标识的回路为 A_1 和 A_2 的相对时间约束回路。若该回路的权值之和为非负数，即：

$$W_{T_RT_{E2}}+W_{T_{E2}T_{S2}}+W_{T_{S2}T_{E1}}+W_{T_{E1}T_{S1}}+W_{T_{S1}T_R}\geqslant0 \tag{7}$$

则说明活动 A_1 和 A_2 满足时间约束的一致性，否则 A_1 和 A_2 之间将发生冲突，需要反馈给用户重新设置时间参数。

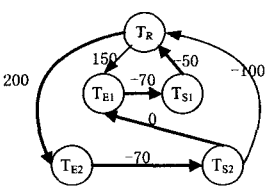
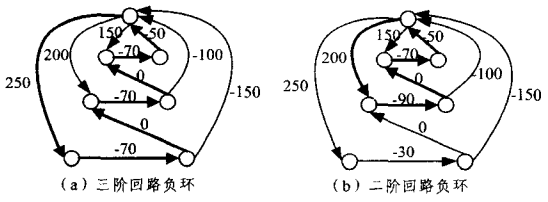


图 7

时间约束网络的引入，给活动规划带来了很大的灵活性。以图 8 所示的两个活动为例，A_1 的时间范围为[50，150]，A_2 的时间范围为[100，200]，按传统规划的方式两者的时间是重叠的，应当判定为冲突。但根据简单时间约束网络负环判定(定理 1)的方法，图中粗线回路的权值之和为 200−70+0−70−50=10，说明 A_1 和 A_2 是满足时间约束的。



(a) 三阶回路负环　(b) 二阶回路负环

图 8　负环示意图

定义 7(n 阶回路)　给定 n 个活动,将这 n 个活动按 T_{ES} 从小到大排列后得到活动序列 $A=\{A_1,A_2,\cdots,A_n\}$,则回路 $\zeta=T_R T_{E_n} T_{S_n} T_{E_{n-1}} T_{S_{n-1}} \cdots T_{E_1} T_{S_1} T_R$ 称作 A 所对应的时间约束网络中的 n 阶回路。

定义 8(n 阶回路的权值)　n 阶回路中包含的所有边的权重之和,称作该 n 阶回路的权值。根据 n 阶回路的定义可知,n 阶回路的权值计算公式为:

$$Value(\zeta)=W_{T_R T_{E_n}}+\sum_{i=1}^{n} W_{T_{E_i} T_{S_i}}+\sum_{i=2}^{n} W_{T_{S_i} T_{E_{i-1}}}+W_{T_{S_1} T_R} \tag{8}$$

若将活动之间的约束条件放宽至两两不发生并行(即活动之间不需要设定最小间隔时间),则对任意 i,有 $W_{T_{S_i} T_{E_{i-1}}}=0$,故式(8)可简化为:

$$Value(\zeta)=W_{T_R T_{E_n}}+\sum_{i=1}^{n} W_{T_{E_i} T_{S_i}}+W_{T_{S_1} T_R} \tag{9}$$

图 7 和图 8 中粗线标识的回路分别为二阶回路和三阶回路。n 阶回路具有以下两条属性。

属性 1　设有 $m(m>n)$个活动,根据定义 7 可知,当 n 个活动确定时,仅有 1 条对应的 n 阶回路,故 m 个活动的 n 阶回路的总数为:

$$C_m^n=\frac{m!}{n!\ (m-n)!} \tag{10}$$

属性 2　设有 $m(m>n)$个活动,即使其所有的 n 阶回路均不为负环,其 $k(k>n)$阶回路和 $j(j<n)$阶回路仍然可能存在负环。

属性 2 可以通过图 8 来说明。图 8(a)中共有 3 条二阶回路,且都为正环,但粗线标识的三阶回路权值之和为 -10,说明这 3 个活动不满足时间约束的一致性。图 8(b)中存在 1 条三阶回路且为正环,但粗线标识的二阶回路的权值之和为 -10,说明前两个活动之间不能满足时间约束的一致性。由此可见属性 2 是成立的。

根据属性 2,可以给出 m 个活动时间约束一致性判定的一般性算法。

GCA(General Checking Algorithm)

给定 m 个活动,对其所有的 $i(i=2,3,4,\cdots,m)$阶回路进行检测,若均为非负环,则这 m 个活动的时间约束一致。

GCA 显然是判断时间约束一致性最直观的方法,但是根据属性 1 可知,要对所有的回路都进行一次检测,系统需要做的检测次数为:

$$C_m^2+C_m^3+\cdots+C_m^{m-1}+C_m^m=2^m-m-1 \tag{11}$$

也就是说,GCA 的时间复杂度是随 m 呈指数型增长的。当 m 取值较大时,GCA 是计算系统无法接受的,因此必须对 GCA 进行改进,以降低计算的复杂度。

4.2　改进算法

GCA 算法虽然直观、明确,但其时间复杂度高的缺点也是明显的。老年人的活动规划并不是一个静态的过程,每当用户加入新的活动,都会导致时间约束网络的变化,需要重新检查时间约束的一致性。

实际上,新添加的活动对已有活动的影响一般都是局部性的。如图 9 所示,当有新的活动 A_{new} 添加进来时,只有 A_{i+1} 和 A_{i+2} 与 A_{new} 的时间发生重叠。由于 A_i 和 A_{i+1} 之间也存在重叠,故 A_{new} 由于会影响 A_{i+1} 而间接影响 A_i。对于 $t<t_1$ 以及 $t>t_2$ 的活动,A_{new} 对它们不产生影响。因此,只要 A_i,A_{i+1},A_{i+2} 和 A_{new} 这 4 个活动的时间约束能够满足一致性,则所有的活动都能够满足时间约束的一致性。

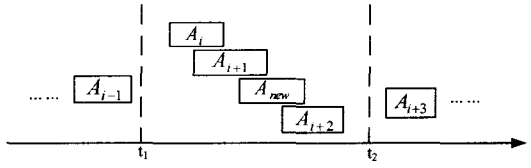


图 9　添加活动后的冲突示意图

根据这一思想,可以对 GCA 进行改进,改进后的算法描述如下。

ICA(Improved Checking Algorithm)

设规划系统中原有 m 个活动,且满足时间约束的一致性,当新添加活动 A_{new} 时,

(1)定义列表 $List_1$,$List_2$,对每个活动 $A_i(i=1,2,\cdots,m)$,检测其是否与 A_{new} 存在时间重叠。如果重叠,则将其加入重叠列表 $List_1$,剩余的活动加入列表 $List_2$。

(2)检测列表 $List_2$ 中的活动是否与重叠列表 $List_1$ 中的活动存在时间重叠。如果存在,将其加入到冲突列表 $List_1$,并将冲突的活动从列表 $List_2$ 中删除;如果不存在冲突,则进入(4)。

(3)返回(2),继续检测冲突活动。

(4)设最终重叠列表 $List_1$ 中共有 k 个活动,将 A_{new} 加入 $List_1$,采用 GCA 算法检测这 $k+1$ 个活动的一致性。

该算法的关键是找到与 A_{new} 的时间存在直接或者间接重叠的活动,将其加入冲突列表 $List_1$。该部分伪代码如下:

```
1:void findLap(){ //将与 Anew 存在直接时间冲突的活动加入到 List1
for(int i=0;i<m;i++)
if((TESnew>=TESi)&&(TESnew<=TDi)||(TDnew>=TESi)&&(TDnew<=TDi))
add Ai to List1; //将 Ai 加入 List1
findMore();
}
2:void findMore(){ //将与 Anew 存在间接时间冲突的活动加入到 List1
int add=0; //设定标记
for(int i=0;i<m;i++)
if(Ai∉List1) //判断 Ai 是否属于 List1
for(int j=0;j<List1.Length;j++){
if((TESi>=TESList1(j))&&(TESi<=TDList1(j))||(TDi>=TESList1(j))&&(TDi<=TDList1(j)))
add Ai to List1; //将 Ai 加入 List1
add++;
  }
If(add>0) //add>0,则递归查找,否则返回
  findMore();
  else return;
}
```

4.3　ICA 与 GCA 算法复杂度比较

下面从所需规划的活动总数以及产生时间重叠的活动的个数两方面来分析 ICA 与 GCA 的性能。实验平台为一台联

想个人电脑，Intel 双核 3.0GHz 处理器，2GB 内存，运行 Windows XP 操作系统。

为了方便对算法性能进行评价，这里给出参数最大重叠数的概念。

定义 9(最大重叠数) 在 ICA 算法中第(4)步运行完后，重叠列表 $List_1$ 最终包含的活动个数，称为最大重叠数。

最大重叠数从本质上来说，就是在新加入活动 A_{new} 后，原有活动中与 A_{new} 产生直接或间接重叠的活动的个数，再加上 A_{new} 本身后的活动总数。例如在图 10 中，当有新的活动 A_{new} 添加进来时，A_{i+1} 和 A_{i+2} 与 A_{new} 的时间直接发生重叠，由于 A_i 和 A_{i+1} 之间也存在重叠，故 A_{new} 与 A_i 的时间间接发生重叠，因此最大重叠数的值为 4。

实验一

第一个实验考察当活动的总数确定而最大重叠数发生变化时 GCA 与 ICA 的计算时间的开销情况。具体实验方案如下。

选取 10 组测试数据，每组数据需要检测的活动的总数均为 10，将第 10 个活动视为新添加的活动，这 10 组数据的最大重叠数依次递增。

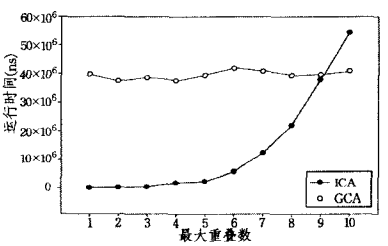


图 10 实验一结果图

由图 10 可知，ICA 的时间开销在大部分情况下是远远低于 GCA 的。仅当最大重叠数的值为 10 时，ICA 的时间开销才大于 GCA。这是因为当最大重叠数为 10 时，ICA 与 GCA 所要检测的回路的个数已经相同了，此时 ICA 算法的第(4)步已经完全等同于 GCA 算法，再加上前 3 步计算开销的，导致计算时间大于 GCA。但是在现实生活中，新添加一个活动后导致所有活动的时间都产生直接或间接重叠的情况是很少见的，因此 ICA 的总体性能是优于 GCA 的。

实验二

第二个实验采用更多的测试数据，分析当活动总数以及最大重叠数均发生变化时 GCA 与 ICA 计算时间的开销情况。

如图 11 所示，当活动的数量增加时，GCA 的开销增加很大。实际上本文 4.1 节中已经指出，GCA 的计算开销是随着活动数量呈指数增长的。

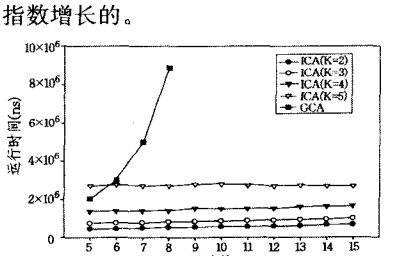


图 11 实验 2 结果图

ICA 的运算开销受活动总数的影响较为有限。图 11 给出了当最大重叠数 K 分别为 2,3,4,5 时 ICA 检测活动一致性的时间开销。根据图 11 可知，当活动数量增加时，ICA 增加的时间开销并不明显，且远远小于 GCA。而当 K 的值增加时，ICA 的时间开销增加是明显的。由此可见 ICA 的计算开销主要取决于最大重叠数 K 的值。

从总体性能上来看，ICA 的性能是优于 GCA 的。因为 ICA 并不关注总共要规划的活动数量，其计算开销主要受限于最大重叠数 K。但现实生活中，新添加活动后导致所有的活动时间发生重叠的情况并不多见，因此 K 的值相对于活动的总数来说不会太大。但由于 GCA 的计算开销与活动总数的关系是呈指数增长的，因此其计算开销总体是大于 ICA 的。举例来说，当活动数量为 8 时，GCA 的时间开销已经达到 8902162ns；而采用 ICA 对 15 个活动进行时间一致性检测，最大重叠数 K 取 5 时，时间开销只有 2708337ns。

结束语 本文主要讨论了老年人生活辅助问题中基于时间约束的活动规划问题。不同人对老年人制定的不同活动会有不同的时间约束，而不同的时间约束会造成活动之间的冲突。本文针对传统的活动时间约束，提出了一种相对宽松的时间约束，并对新的时间约束中活动时间冲突检测问题进行了分析。在此基础上，介绍了时间约束网络的相关概念和理论，利用简单时间约束网络来描述本文提出的活动时间约束方法，将活动时间冲突检测的问题转化为简单时间网络求解一致性的问题。针对简单时间约束网络一致性的检测，本文介绍了一般性的检测算法，分析了其时间复杂度，并对该算法进行了改进。改进算法将与新加入活动存在直接或间接时间冲突的活动记录起来，不去考虑其他活动与新加入活动的时间冲突问题，进而降低了冲突检测算法的复杂度。本文最后对传统算法和改进后的算法在所需规划的活动总数以及产生时间重叠的活动的个数两方面进行了性能评估。由实验结果可以看出，改进算法在多数情况下简化了活动规划中的冲突检测问题。

参 考 文 献

[1] http://www.cogknow.eu/overview

[2] Cesta A, Cortellessa G, Pecora F, et al. Supporting Interaction in the ROBOCARE Intelligent Assistive Environment[C]// AAAI 2007 Spring Symposium. Interaction Challenges for Intelligent Assistants, Stanford University, CA, USA, March 2007

[3] Pecora F, Cesta A. DCOP for Smart Homes: a Case Study[J]. Computational Intelligence, 2007, 23(4): 395-419

[4] Mccarthy C E. A plan-based personalized cognitive orthotic[C]// Proceedings of the 6th International Conference on AI Planning and Scheduling

[5] Pollack M E, Brown L E, Colbry D, et al. Autominder: an intelligent cognitive orthotic system for people with memory impairment[J]. Robotics and Autonomous Systems, 2003, 44(3/4): 273-282

[6] Rudary M, Singh S, Pollack M E. Adaptive Cognitive Orthotics: Combining Reinforcement Learning and Constraint-based Temporal Reasoning[C] // International Workshop Then Conference. 2004: 719-726

[7] Oddi A, Cesta A. Incremental forward checking for the disjunctive temporal problem[C]//14th European Conference on Artificial Intelligence. IOS Press: 108-112

[8] Stergiou K, Koubarakis M. Backtracking algorithms for disjunctions of temporal constraints[J]. Artificial Intelligence, 120: 81-117

[9] Dechter R, Meiri I, Pearl J. Temporal constraint net work[J]. Artificial Intelligence, 1991, 49(1-3): 61-95

[10] 徐瑞；徐晓飞，崔平远. 基于时间约束网络的动态规划调度算法[J]. 计算机集成制造系统，2004(2)