基于3D碎裂度布局策略的可重构硬件任务调度算法

徐金甫 刘 露 李 伟* 南龙梅
(解放军信息工程大学 郑州 450001)

摘 要：现有硬件任务调度算法任务描述不完善且忽视时间维上紧凑性。该文考虑任务下载时间、完善任务属性，以器件2维资源与时间建立3维资源模型，将任务布局问题抽象成特殊的3维空间放置问题，在此模型上分析出现有算法不能克服任务不可预知性和资源占用多变性，导致调度成功率和资源利用率低。针对此问题，该文提出了一种3维可重构任务调度算法3D_RTSA。设计并实现了基于任务紧迫度的调度策略和基于3D碎裂度的布局策略。与其他4种算法实验对比结果表明，在重负载、小任务C30情况下，3D_RTSA调度成功率比GC, Look-aheadest, SPSA, DTI算法分别高3%, 21%, 28%, 35%左右；在轻负载、大任务C50情况下，资源利用率比Look-aheadest, SPSA算法分别高5%, 18%左右，且该文算法时间复杂度并未增加。

关键词：3维资源模型；任务紧迫度；3D碎裂度；调度成功率；资源利用率
中图分类号：TP316.4 **文献标识码**：A **文章编号**：1009-5896(2018)08-2020-08
DOI: 10.11999/JEIT171205

Reconfigurable Hardware Task Scheduling Algorithm Based on 3D Fragmentation Layout Strategy

XU Jinfu LIU Lu LI Wei NAN Longmei
(*The PLA Information Engineering University, Zhengzhou* 450001, *China*)

Abstract: The existing hardware task scheduling algorithms describe task imperfectly and ignore the compactness of time dimension. The task downloading time is considered for improving the task attribute, and the 3D-resource model with the two dimensional resource of device and time is established, in order to abstract the issue of task layout into a special three-dimensional space placement issue. With this model, it is concluded that the existing algorithms can not overcome the unpredictability of the task and the diversity of resource occupancy, leading low scheduling success rate and resource utilization rate. To solve the problem, a three dimensional reconfigurable task scheduling algorithm called 3D_RTSA is proposed. A scheduling strategy based on task urgency and a layout strategy based on 3D fragmentation are designed and implemented. Compared with the other 4 algorithms, the results show that the scheduling success rate of 3D_RTSA is 3%, 21%, 28%, 35% higher than that of GC, Look-aheadest, SPSA and DTI algorithms under the condition of heavy load and small task C30, and the utilization ratio of resources is 5% and 18% higher than that of Look-aheadest and SPSA algorithm under the condition of light load and large task C50. Besides, the time complexity of the algorithm is not increased.

Key words: 3D-resource model; Task urgency; 3D fragmentation; Scheduling success rate; Resource utilization

1 引言

近年来，针对不同资源模型(1D, 2D)和任务属性(实时，非实时)，国内外学者提出了许多硬件任务调度算法[1−4]，从而衍生出不同的调度策略(FCFS[5]，资源预约[6]等)、布局策略(FF[7]，FE[8]，MAS[9]等)和可重构资源管理方法[10,11]。文献[12]提出了一种亚可抢占调度算法SPSA，将大区域分割成多个子区域，进行递归扫描，该算法时间开销会随着子区域数量的增加而大幅增加。以执行时间最短优先调度的调度策略存在极大的不合理性，这可能降低调度成功率。文献[13]提出了一种分组-邻接边调度算法GC，按任务的宽高比将任务分为垂直任务和水平任务，分搜索两个坐标方向寻找可放置点，减少了一半的匹配时间开销。该算法虽然考虑

收稿日期：2017-12-21；改回日期：2018-05-18；网络出版：2018-06-06
*通信作者： 李伟 liulu13213238773@163.com
基金项目：国家自然科学基金(61404175)
Foundation Item: The National Natural Science Foundation of China (61404175)

时间，但只是将时间作为一个约束条件，实质还是2维放置问题，且FCFS调度策略过于简单，可能导致后续紧迫度高的任务没有足够资源而被拒绝[14]。文献[15]提出了一种非抢占、实时硬件任务调度算法CA-MAE。该算法考虑硬件通信开销，将MAS策略与通信系数α相结合，作为布局决策函数，能减少通信开销和提高资源利用率。但该算法和文献[12]一样，采用FCFS调度策略，只考虑2维空间资源的紧凑性，使单位时间内的资源利用率降低。

综上分析，现有硬件任务调度算法各有优势，但也有严重缺陷。本文完善任务模型，考虑任务下载时间；完善资源模型，合并器件2维资源和时间维资源，建立3维可重构资源模型，将任务的布局问题抽象为3维空间布局问题。将现有算法在3维模型上应用，发现其不能克服实时任务的不可预知性和资源占用多变性，导致调度成功率和资源利用率不高。本文针对这些问题，提出了基于任务紧迫度的调度策略和基于3D碎裂度的布局策略，提高了调度成功率和资源利用率，降低了时间复杂度。

2　模型建立

2.1　配置任务调度系统结构模型

图1为可重构阵列[16]的任务调度系统结构模型(FPGA类似)。主要包括配置任务调度系统和可重构阵列。调度器负责接收新任务，并通过调度算法实现任务间的有序调度和任务状态的正确转换。布局器负责为任务在有限的2维资源和无限的时间维资源中找到最佳放置位置，确定启动时间。资源管理器负责记录和及时更新可重构阵列资源占用情况，并向布局器反馈。加载器负责将已布局硬件任务下载至阵列内完成配置。

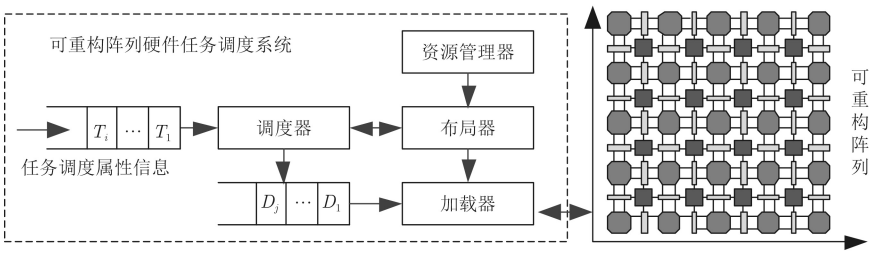


图 1 硬件任务调度系统结构模型

2.2　可重构任务模型

在3维空间中，将硬件任务看作固定大小的长方体块，用$T(w, h, e, a, d)$描述，w和h分别代表任务块长和宽，e是任务执行时间，也是任务块的高，a是任务到达调度系统时间，d是系统为任务规定的截止时间。完善任务属性及优先级后，任务块用$A(w, h, e, a, v, d, p)$描述，v表示任务块下载时间，p表示任务调度优先级。任务成功布局后用$D(x_1, y_1, x_2, y_2, s, f)$描述，$(x_1, y_1)$表示任务块所占矩形器件资源左下角RCU的坐标，$(x_2, y_2)$表示右上角RCU的坐标，$s$表示任务开始执行时间，$f$表示任务结束时间。任务$A_1(4, 5, 5, 1, 2, 10, 1)$经过成功布局后得到$D_1(0, 0, 4, 5, 3, 8)$，$D_1$在3维空间放置如图2所示。

为更好地理解下文，现做如下定义。

定义 1　最早可能开始执行时间$\mathrm{Ear_S}_{i+1}$。指布局算法能为任务A_{i+1}布局提供的最低时间切面对应的时间，$A_i.s^m$表示任务A_i放置在角点m后确定的开始时间。

定义 2　最早可能结束时间$\mathrm{Ear_F}_{i+1}$，由$\mathrm{Ear_S}_{i+1}$确定。

定义 3　最低布局空间$\mathrm{3D_Lay}_{i+1}(W, H, \mathrm{Ear_S}_{i+1}, \mathrm{Ear_F}_{i+1})$。指从$\mathrm{Ear_S}_{i+1}$起，长为$W$，宽为$H$，高为$A_{i+1}.e$的3维空间，与角点$m$位置有关。

2.3　模型分析

问题 1　假设可重构阵列状态矩阵为$\boldsymbol{R}(12, 8)$，有4个任务等待调度，分别是$T_1(4, 5, 5, 1, 8)$，$T_2(5, 2, 2, 2, 9)$，$T_3(4, 5, 5, 3, 11)$和$T_4(9, 4, 5, 8, 14)$，文献[17]对这4个任务的调度和布局结果如图3(a)所示，T_4被丢弃。若将任务T_3放置到图3(b)所示的位置，则T_4成功被调度；若第4个任务为T_4' 7,

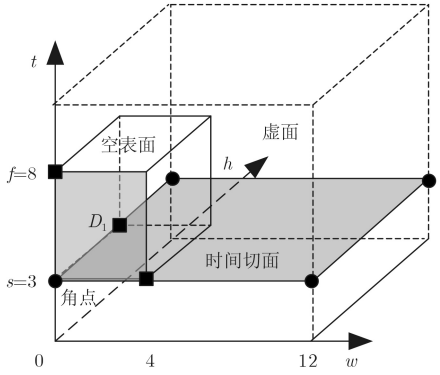


图 2 任务块在3维空间布局示意图

4, 5, 7, 13)，则$T_4{}'$在图3(a)布局后成功被调度，如图3(c)所示，而在图3(b)布局后会由于不能在截止期内结束任务而被丢弃。

问题 2 文献[14]提出基于面积与紧迫度相乘的调度策略，明显优于FCFS调度策略。按文献[14]调度策略完善任务T_1(4, 5, 5, 1, 8)，T_2(5, 5, 2, 2, 9)，T_3(4, 5, 5, 3, 13)和T_4(8, 5, 5, 3, 11)后，得到A_1(4, 5, 5, 1, 2, 10, p_1=0.198)，A_2(5, 2, 2, 2, 1, 10, p_2=0.112)，A_3(4, 5, 5, 3, 2, 15，p_3=0.088)和A_4(8, 5, 5, 3, 4, 15, p_4=0.083)。由于$p_1>p_2>p_3>p_4$，所以调度次序是A_1，A_2，A_3和A_4。如图4(a)所示，A_4被丢弃。当先A_4后A_3调度和布局时，A_3和A_4都被成功调度。

问题1说明文献[17]中最大邻接面布局策略并不能很好地解决布局问题，同时也说明布局策略的固定性不能适应后续任务属性的不可预知性。问题2说明固定调度策略不能适应资源占用情况的多变性。这两个问题说明调度与布局之间只有存在“信息反馈”，可以灵活调整，才能克服实时任务属性不可预知性和资源占用情况多变性，最终提高调度成功率。

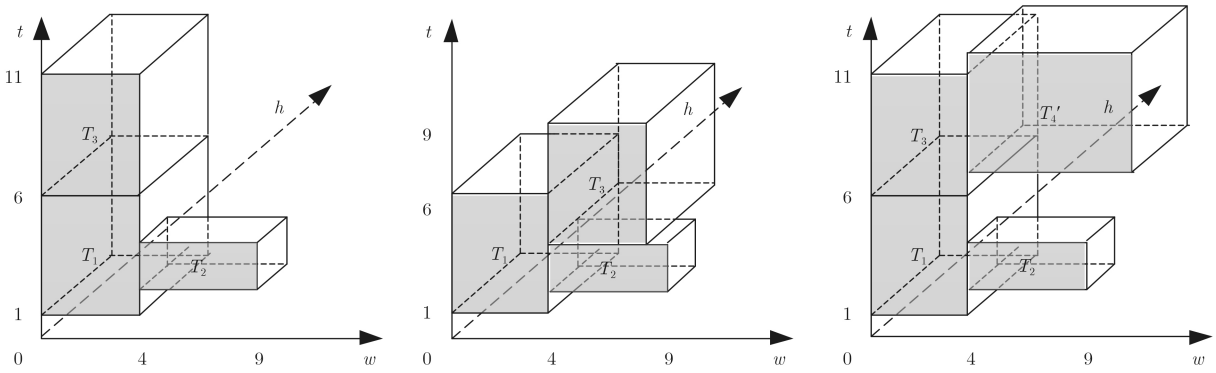
3 3D_RTSA调度算法

3.1 基于配置任务紧迫度的调度决策函数

文献[18]已证明优先放置面积较小的任务更有利于提高任务调度成功率。文献[14]关注了面积和紧迫度，但由于该决策函数并不能平衡它们的权重，出现了问题2所描述的问题，且紧迫度定义并不完善。本文将任务体积“转化”为下载时间v，融入紧迫度。配置任务T_i调度优先级决策函数(紧迫度)$p(i)$为

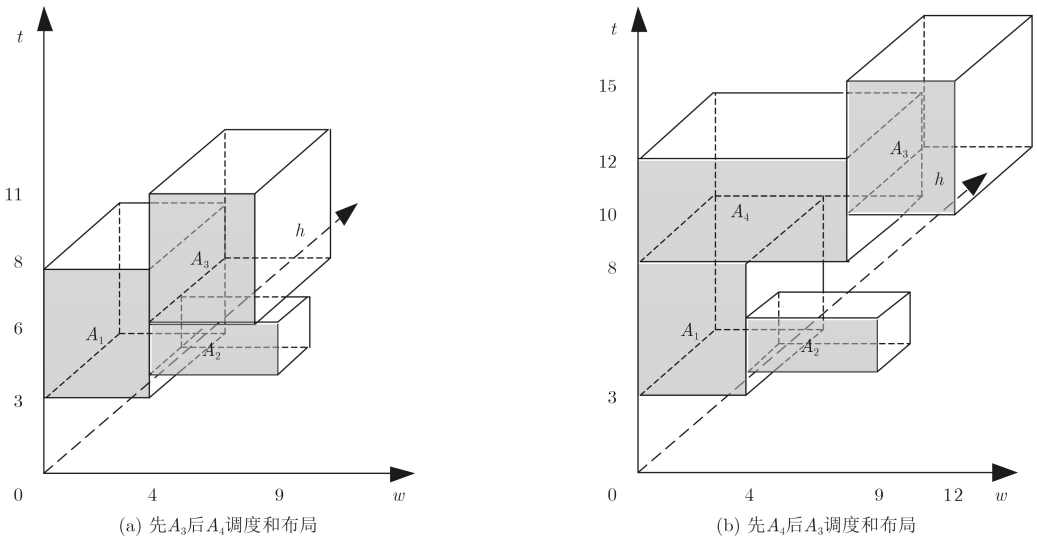
$$p(i) = A_i.d - A_i.e - A_i.v + 1 \tag{1}$$

从式(1)可知，任务体积与紧迫度并不是如文献[13]一样是并列、并重关系，而是递进关系，体积小的任务可能优先调度，但最终决策函数是紧迫度。当两个任务紧迫度相同时，选择体积小的任务优先调度。



(a) 最大邻接面调度结果 (b) 非最大邻接面调度结果 (c) T_4换成T_4'调度结果

图 3 3种情况调度结果对比图



(a) 先A_3后A_4调度和布局 (b) 先A_4后A_3调度和布局

图 4 文献[14]的优先级调度策略实例

3.2 算法描述

3D_RTSA算法包含调度算法和布局算法。其主要思想是：当 SS_List中有多任务时，需考虑后续任务对当前任务布局的影响；当SS_List中只有1个任务时，为满足实时性，应及时处理，无需考虑后续任务影响；当多任务布局失败，对其进行深度优化布局，所以布局算法包括两种不同的布局策略。调度与布局需要相互反馈，3D_RTSA算法流程如图5所示。链表TR_List用于缓存刚到调度系统的任务；链表SS_List用于缓存完善后的任务；链表PS_List用于缓存布局后的任务；链表PCL用于缓存Ear_S_{i+1}之后的所有角点；链表ACL用于缓存Ear_S_{i+1}之后可以放置该任务的角点。

4 3D_RTSA布局策略

4.1 基于3D碎裂度的双任务布局策略

由于硬件任务在3维空间布局存在特殊性，使任务放置存在规律性：(1)趋角性。无论是利用吸引子法作为布局决策函数[19]，还是其他布局策略，都有趋角放置的规律。(2)A_{i+1}和A_i放置位置存在3种情况。如图6所示。位置固定性给设计双任务布局策略带来启发：A_i放置后，能在A_{i+1}最低布局空间3D_Lay_{i+1}(W, H, Ear_S_{i+1}, Ear_F_{i+1})中留最多、最整的3维空闲空间，且保证两任务都在各自截止期内能结束。

为判断A_i最佳放置位置，参考文献[8]平面碎裂

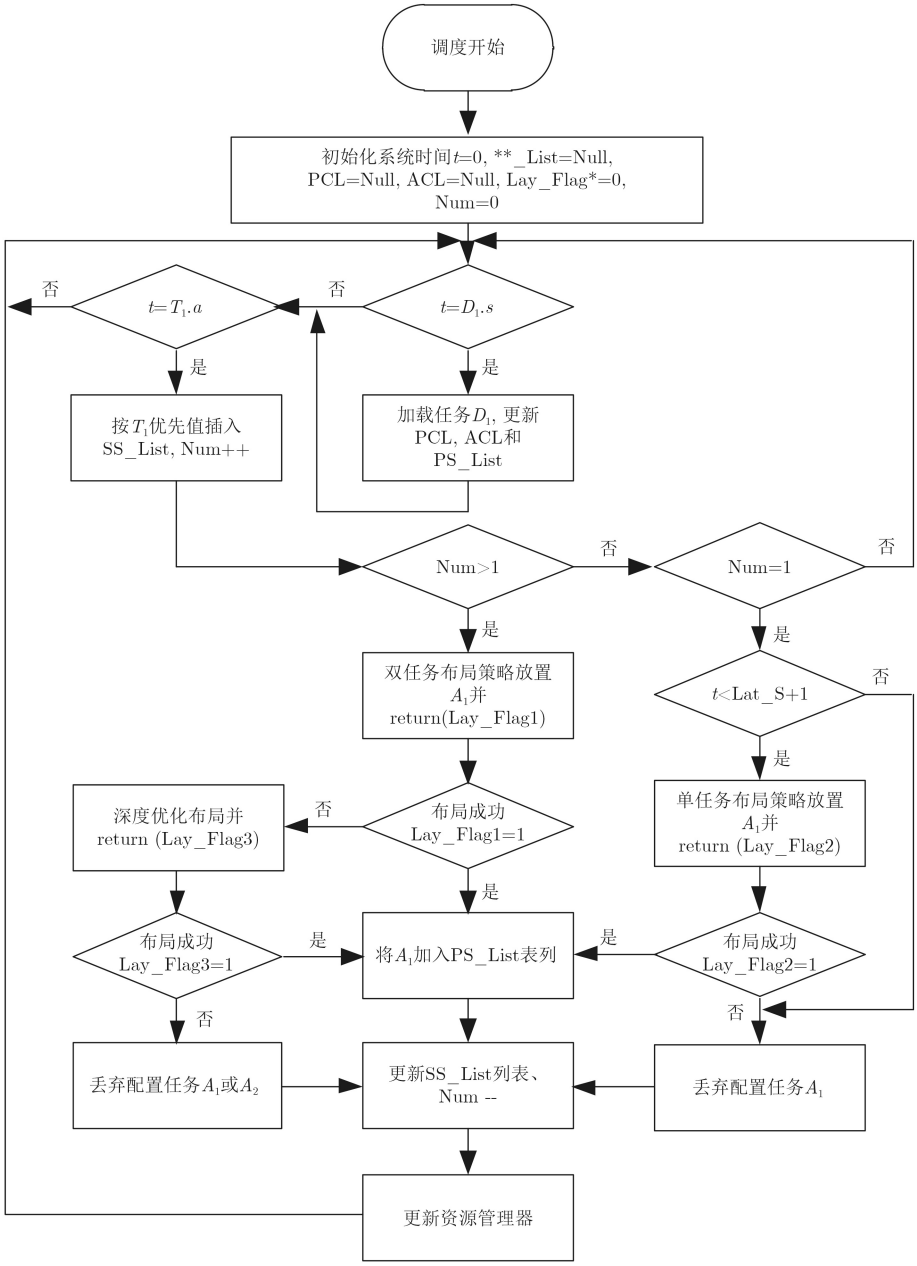


图 5 3D_RTSA算法流程

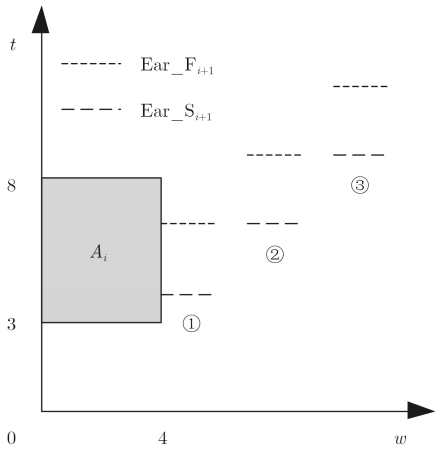


图 6 前后硬件任务相对放置位置情况

度计算方法，本文提出基于表面积的3维空间碎裂度布局决策函数。如图7所示(属于情况②)，时间切面t=10与A_3任务块相交形成“孤岛”，Ear_S$_4$ = 10，以此图为模型，A_i在角点m放置后，A_{i+1}最低布局空间3D_Lay$_{i+1}$中空闲空间表面积计算方法如式(2)所示(两个灰色时间切面与4个虚面围成的3维空间即是A_4最低布局空间)。

$$\left.\begin{aligned} &3\mathrm{D}_2\_\mathrm{S}_{i+1}^m = S_{i+1}^m + d_{\min}\cdot\Delta t\cdot g + \sum_{h=1}^{4} S_{i+1}^h \\ &\Delta t = \begin{cases} A_{i+1}.e, & \text{情况1} \\ A_i.f^{\,m} - \mathrm{Ear\_S}_{i+1}, & \text{情况2} \\ 0, & \text{情况3} \end{cases} \end{aligned}\right\} \tag{2}$$

S_{i+1}^m表示未布局A_i时，3D_Lay$_{i+1}$空闲空间的表面积；$\sum S_{i+1}^h$表示布局A_i后，3D_Lay$_{i+1}$增加的空闲空间表面积，即“孤岛”4个侧空表面面积之和，与$d_{\min}$和Δt有关；$d_{\min}\cdot\Delta t\cdot g$表示“惩罚面积”，$d_{\min}$表示“孤岛”离虚面或空表面最近距离，$\Delta t$表示“孤岛”的高，$g$表示“惩罚因子”，本文取$g=2$。

表面积为3D$_2$_S$_{i+1}^m$的正方体体积计算方法如式(3)所示。在角点m放置A_i后，3D_Lay$_{i+1}$空闲空间体积计算方法如式(4)：

$$3\mathrm{D}_2\_\mathrm{V} = \left(3\mathrm{D}_2\_\mathrm{S}_{i+1}^m/4\right)^{3/2} \tag{3}$$

$$3\mathrm{D}_2\_\mathrm{V}_{i+1}^m = V_{i+1}^m - \Delta t\cdot A_i.w\cdot A_i.h \tag{4}$$

V_{i+1}^m表示未布局A_i时，3D_Lay$_{i+1}$空闲空间的体积；$\Delta t\cdot A_i.w\cdot A_i.h$表示在角点$m$放置$A_i$后，3D_Lay$_{i+1}$减少的空闲空间体积。在角点$m$放置$A_i$后，3D_Lay$_{i+1}$周正度、碎裂度计算方法分别如式(5)，式(6)：

$$3\mathrm{D}_2\_\mathrm{Q}_{i+1}^m = 3\mathrm{D}_2\_\mathrm{V}_{i+1}^m/3\mathrm{D}_2\_\mathrm{V} \tag{5}$$

$$\begin{aligned} 3\mathrm{D}_2\_\mathrm{F}_{i+1}^m &= 1 - 3\mathrm{D}_2\_\mathrm{Q}_{i+1}^m \\ &= 1 - \frac{8\cdot\left(V_{i+1}^m - \Delta t\cdot A_i.w\cdot A_i.h\right)}{\left(S_{i+1}^m + d_{\min}\cdot\Delta t\cdot g + \sum_{h=1}^{4} S_{i+1}^h\right)^{3/2}} \end{aligned} \tag{6}$$

其中，$d_{\min}$，V_{i+1}^m，S_{i+1}^m和$\sum S_{i+1}^h$与角点有关，Δt与A_{i+1}属性有关，即碎裂度不仅与放置角点有关，也与后续任务A_{i+1}属性有关。布局策略会选择在3D_Lay$_{i+1}$中产生最小碎裂度的角点放置A_i，当有多个碎裂度相同的角点时，选择最低角点放置A_i。

单任务布局策略是将后续任务属性虚拟化，只要在最差的情况下使虚拟化的3D_Lay$_{i+1}$(A_i执行阶段所占的3维空间)碎裂度最小，就能保证在A_{i+1}布局时，可以获得最多、最完整的3维空闲空间。

4.2 深度优化布局策略

结合2.3节问题2和4.1节双任务布局策略，可以分析出导致其调度失败原因有两个：(1)A_i和A_{i+1}相对位置固定，无法缩小两任务执行的总时间。(2)A_i布局后，导致3D_Lay$_{i+1}$碎裂度高，在截止时间与3维空间约束下无法成功放置A_{i+1}。双任务布局失败表示在问题2上已不能进一步优化，只有尝试解决问题1，调换两个任务的调度与布局顺序作为深度优化布局策略是合理、可行的。但盲目深度优化布局会增加调度时间开销，需提前知晓优化的必要性。如图8所示，图8(a)是问题2中A_4调度失败的情况，图8(b)是A_3和A_4都调度成功的情况，明显看出调序后两任务执行总时间减小。式(7)是深度优化布局的必要性条件。

$$A_{i+1}.e + A_{i+1}.v > A_i.e + A_i.v \tag{7}$$

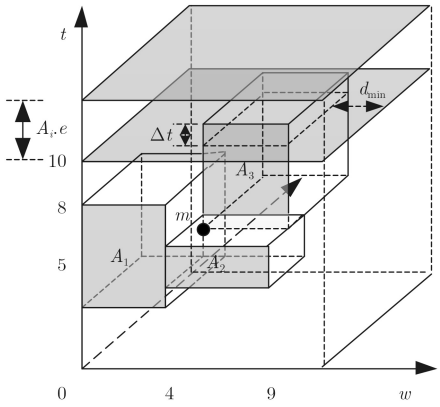


图 7 双任务布局时3D碎裂度计算模型

5 仿真实验及结果分析

5.1 仿真实验环境与实验方案

为了方便对比，参照文献[7]实验模型，可重构

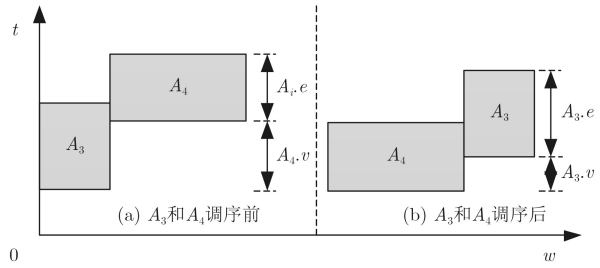


图 8 深度优化布局必要性分析

器件按照Xilinx XCV 1000规模定义，具有96×64个RCU。将任务按照面积分为3类，分别记作C30, C40和C50, Cn的长和宽在[5, n]内均匀分布。任务运行时间e在[5, 50]个时间单位内均匀分布，任务到达时间a在[1, 100]个时间单位均匀分布，松弛时间Laxity在区间A, B和C内均匀分布，A, B, C分别对应[1, 50], [50, 100]和[100, 200]个时间单位。

仿真实验过程：首先在各自指定的区间生成一组任务的长、宽、执行时间和松弛时间，然后根据指定的负载率L计算出测试集中最后一个任务的截止期，并以此确定相邻任务间的到达间隔时间，即到达速率，从而确定到达时间a。为更好分析调度算法优劣和更好地与其他算法对比，本文将负载率分别取0.3, 0.5, 0.7, 1.0, 1.5和2.0。在不同任务面积Cn，负载率L和松弛时间Laxity下，各生成100组任务测试集，每组任务包含1000个随机生成的任务。

5.2 实验结果及性能分析

用松弛时间为B:[50, 100]、3种任务面积Cn生成的3种不同任务测试集，在不同器件负载率下，测试SPSA[12], GC[13], DTI[14], Look-aheadest[17]和本文算法的调度成功率，结果如图9所示。

从图9可以看出，本文3D_RTSA算法相对于Look-aheadest, DTI, GC和SPSA算法在调度成功率上都有提高。调度成功率并不是评价调度算法的唯一标准，资源利用率也非常关键。本文用松弛时间为B:[50, 100]、3种任务面积Cn生成的3种不同任务测试集，在负载率$L=1$下，测试Look-aheadest, SPSA和本文算法的资源利用率，结果如图10所示。

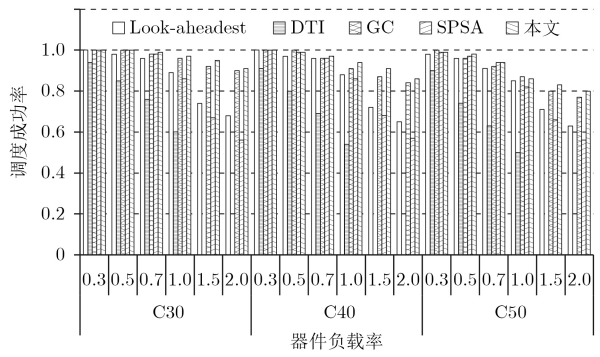


图 9 5种调度算法调度成功率对比

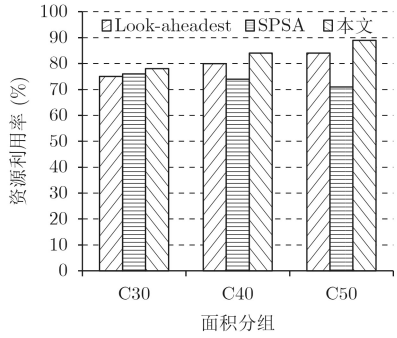


图 10 3种调度算法资源利用率对比

从图10可以看出，在本文算法相对于Look-aheadest和SPSA算法资源利用率更高。因为本文算法从设计初就重视任务在时间维的紧凑性问题，且调度成功率又高于两种算法，资源利用情况自然更好。

对Look-aheadest, DTI, GC, SPSA和本文算法的时间复杂度进行分析与对比，结果如表1所示。其中，E表示正在执行的任务个数；R表示算法预约任务的个数，在本文算法中对应链表PS_List中的任务个数；N表示一组任务测试集中的任务个数，本文N=1000；G与器件规模有关，$G=W\times H$。从表1可以看出，本文的算法时间复杂度与DTI算法持平，比Look-aheadest, GC和SPSA低。

表 1　5种算法时间复杂度对比

算法	调度算法复杂度	布局算法复杂度
Look-aheadest	$O(E2R+G)$	$O(E2R)$
DTI	$O(N)$	$O(E+R)$
GC	$O(N(E+R))$	$O(E+R)$
SPSA	$O(R2)$	$O(GR)$
本文	$O(N)$	$O(E+R)$

6　结束语

针对现有可重构调度算法未考虑任务下载时间、未重视时间维上布局紧凑性、调度与布局策略不合理性等问题，本文以问题指导设计，首先完善任务模型，直接在3维空间资源模型上分析现有调度与布局算法缺点，并得出重要结论，以指导本文3D_RTSA调度算法和布局算法的设计。提出了基于时间紧迫度的调度策略、基于3D碎裂度的布局策略和优化策略。之后设计仿真实验系统对3D_RTSA, Look-aheadest, DTI, GC和SPSA算法在调度成功率、资源利用率、时间复杂度等进行测试对比。实

验结果表明，3D_RTSA算法在没有增加时间开销情况下，调度成功率和资源利用率都优于其他4种算法。

本文3D_RTSA算法只解决了相邻任务间由于任务属性不可预知性带来的问题，可以针对3个任务或者更多任务，对其扩展设计，时间开销会有所增加，但调度成功率也会增加。链表ACL里存在许多冗余可放置角点，消除后会减小算法时间开销，在后续工作中会继续完善。

参 考 文 献

[1] 张晶, 孙少杰, 范洪博, 等. 高实时性异构多核处理器任务调度算法[J]. 计算机工程, 2017, 43(5): 55–59. doi: 10.3969/j.issn.1000-3428.2017.05.009.
ZHANG Jing, SUN Shaojie, FAN Hongbo, *et al*. Task scheduling algorithm in heterogeneous multi-core processor with high real-time performance[J]. *Computer Engineering*, 2017, 43(5): 55–59. doi: 10.3969/j.issn.1000-3428.2017.05.009.

[2] AL-WATTAR A, AREIBI S, and GREWAL G. An efficient evolutionary task scheduling/binding framework for reconfigurable systems[J]. *International Journal of Reconfigurable Computing*, 2016(2): 1–24. doi: 10.1155/2016/9012909.

[3] MOLLAJAFARI M and SHAHHOSEINI H S. An Efficient ACO-based Algorithm for Scheduling Tasks onto Dynamically Reconfigurable Hardware Using TSP-likened Construction Graph[M]. New York: Kluwer Academic Publishers, 2016: 695–712. doi: 10.1007/s10489-016-0782-2.

[4] 徐晓东. 动态可重构系统中任务调度与布局算法研究[D]. [硕士论文], 中国科学技术大学, 2017: 25–59.
XU Xiaodong. Task scheduling and floorplanning algorithm in dynamically reconfigurable systems[D]. [Master dissertation], University of Science and Technology of China, 2017: 25–59.

[5] 周学功, 梁楔, 黄勋章, 等. 可重构系统中的实时任务在线调度与放置算法[J]. 计算机学报, 2007, 30(11): 1901–1909. doi: 10.3321/j.issn:0254-4164.2007.11.002.
ZHOU Xuegong, LIANG Liang, HUANG Xunzhang, *et al*. On-line scheduling and placement of real-time tasks for reconfigurable computing system[J]. *Chinese Journal of Computers*, 2007, 30(11): 1901–1909. doi: 10.3321/j.issn:0254-4164.2007.11.002.

[6] STEIGER C, WALDER H, and PLATZNER M. Heuristics for Online Scheduling Real-time Tasks to Partially Reconfigurable Devices[M]. Springer Berlin Heidelberg, 2003: 575–584. doi: 10.1007/978-3-540-45234-8_56.

[7] STEIGER C, WALDER H, and PLATZNER M. Operating systems for reconfigurable embedded platforms: Online scheduling of real-time tasks[J]. *IEEE Transactions on Computers*, 2004, 53(11): 1393–1407. doi: 10.1109/TC.2004.99.

[8] SEPTIEN J, MOZOS D, MECHA H, *et al*. Perimeter quadrature-based metric for estimating FPGA fragmentation in 2D HW multitasking[C]. IEEE International Symposium on Parallel and Distributed Processing, Miami, USA, 2008: 1–8. doi: 10.1109/IPDPS.2008.4536508.

[9] 齐骥, 李曦, 胡楠, 等. 基于硬件任务顶点的可重构系统资源管理算法[J]. 电子学报, 2006, 34(11): 2094–2098. doi: 10.3321/j.issn:0372-2112.2006.11.034.
QI Ji, LI Xi, HU Nan, *et al*. Algorithms of resource management for reconfigurable systems based on hardware task vertexes[J]. *Acta Electronica Sinica*, 2006, 34(11): 2094–2098. doi: 10.3321/j.issn:0372-2112.2006.11.034.

[10] LEUNG Y T, TAM T W, WONG C S, *et al*. Packing squares into a square[J]. *Journal of Parallel & Distributed Computing*, 1990, 10(3): 271–275. doi: 10.1016/0743-7315(90)90019-L.

[11] HANDA M and VEMURI R. An efficient algorithm for finding empty space for online FPGA placement[C]. Design Automation Conference. San Diego, USA, 2004: 960–965. doi: 10.1145/996566.996820.

[12] 许新达. 基于局部可重构计算的在线硬件任务调度算法研究[D]. [硕士论文], 湖南大学, 2010.
XU Xinda. The research on online hardware task schedule algorithm based-on partial reconfigurable computing[D]. [Master dissertation], Hunan University, 2010.

[13] 刘彦, 李仁发, 许新达, 等. 一种异构可重构片上系统的实时任务调度算法[J]. 计算机研究与发展, 2010, 47(6): 1116–1124.
LIU Yan, LI Renfa, XU Xinda, *et al*. Research on a real-time task scheduling algorithm for hybrid reconfigurable system-on-chip[J]. *Journal of Computer Research and Development*, 2010, 47(6): 1116–1124.

[14] 曹晓磊. 基于LRSS的可重构任务调度算法研究[D]. [硕士论文], 解放军信息工程大学, 2010: 27–37.
CAO Xiaolei. Research on reconfigurable task scheduling algorithms based on LRSS[D]. [Master dissertation], PLA Information Engineering University, 2010: 27–37.

[15] SHENG Y, LIU Y, LI R, *et al*. A communication-aware scheduling algorithm for hardware task scheduling model on FPGA-based reconfigurable systems[J]. *Journal of Computers*, 2014, 9(11). doi: 10.4304/jcp.9.11.2552-2558.

[16] 杨锦江, 曹鹏, 杨军. 面向分组密码算法的高面积效率可重构架构[J]. 东南大学学报(自然科学版), 2016, 46(5): 939–944. doi: 10.3969/j.issn.1001-0505.2016.05.007.
YANG Jinjiang, CAO Peng, and YANG Jun.

Reconfigurable architecture with high area efficiency for block cipher algorithms[J]. *Journal of Southeast University* (*Natural Science Edition*), 2016, 46(5): 939–944. doi: 10.3969/j.issn.1001-0505.2016.05.007.

[17] 彭日光. 动态可重构片上系统的任务在线放置和调度算法研究[D]. [硕士论文], 湖南大学, 2009: 15–23.
PENG Riguang. Research on algorithms of onling placement and scheduling of real-time tasks for reconfigurable system-on-chip[D]. [Master's dissertation], Hunan University, 2009: 15–23.

[18] BANERJEE S, BOZORGZADEH E, and DUTT N. Physically-aware HW-SW partitioning for reconfigurable architectures with partial dynamic reconfiguration[C]. Design Automation Conference. Anaheim, USA, 2005: 335–340. doi: 10.1145/1065579.1065667.

[19] 王金敏. 三维矩形布局吸引子性质的研究[J]. 图学学报, 2016, 37(3): 355–358. doi: 10.11996/JG.j.2095-302X.2016030355.
WANG Jinmin. Research on the property of attractive factor in three dimensional rectangular packing problems[J]. *Journal of Graphics*, 2016, 37(3): 355–358. doi: 10.11996/JG.j.2095-302X.2016030355.

徐金甫：男，1965年生，教授，硕士生导师，研究方向为专用集成电路设计.

刘　露：男，1992年生，硕士，研究方向为专用集成电路设计.

李　伟：男，1983年生，副教授，硕士生导师，研究方向为安全专用芯片设计.

南龙梅：女，1981年生，讲师，研究方向为安全专用芯片设计.