第 8 章　微系统划分方法

鸢飞戾天，鱼跃于渊《诗经·大雅·旱麓》

微系统是由若干模块组成的一个板上系统，每个模块可以有多核处理器，包括异构多核，模块间通过通信完成数据传输，每个模块完成若干任务，每个任务可以软件实现也可以硬件实现。微系统划分是将整个系统的 N 个任务划分成 m 个模块，使得模块间的通信代价最小，同时在每个模块内依据任务性能进行软硬件（多核）划分实现性能最优，依据这些模块和软硬件（多核）划分计算整个系统的性能最优。最后确定的模块和软硬件（多核）划分结果成为这个系统的综合解决方案。

本章介绍三类微系统化方法：基于模块的微系统划分方法、面向多核的微系统划分方法、以及基于遗传算法的微系统划分算法。

第 8.1 节 基于模块的微系统划分

本节介绍基于模块的微系统划分方法，是将第 5 章多目标划分方法和第 7 章多模块划分方法组合在一起，进行微系统划分，首先基于通信代价将整个系统进行模块化，然后对每个模块依据任务的多维属性进行软硬件划分，构建基于模块的微系统划分方法。

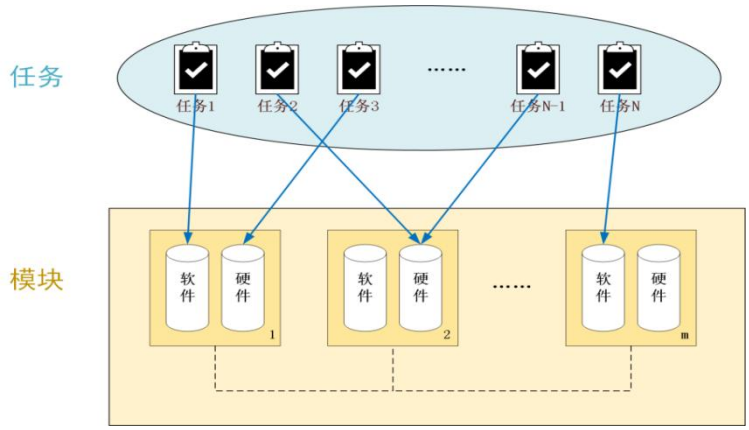

图 8-1　微系统划分示意图[40]

8.1.1. 划分算法

划分算法 8.1 的示意图如图 8-2 所示。

算法 8.1 基于模块的微系统划分划分算法

Input: 含 n 个节点的{t_1，t_2，…，t_n}的有权无向图 G， A=(C_{ij})是节点间通信代价邻接矩阵，M 模块个数，EN 模块包含元素最大个数

Output: 基于模块的微系统划分结果

第 1 步：调用多模块划分算法，依据通信代价和模块数设定，将系统的任务划分到模块中，计算模块间的通信代价以及整个微系统的通信代价，通常要求微系统的通信代价最少；

第 2 步：调用多目标划分算法，将每个模块内依据任务性能属性进行软硬件划分，计算出每个模块的属性；

第 3 步：依据微系统的通信代价和每个模块属性，计算出整个系统的性能，求出任务划分和模块设定方案，通常这个不是最优解，重复第一步或第二步，并进行比较，给出相对较优方案；

第 4 步：算法结束

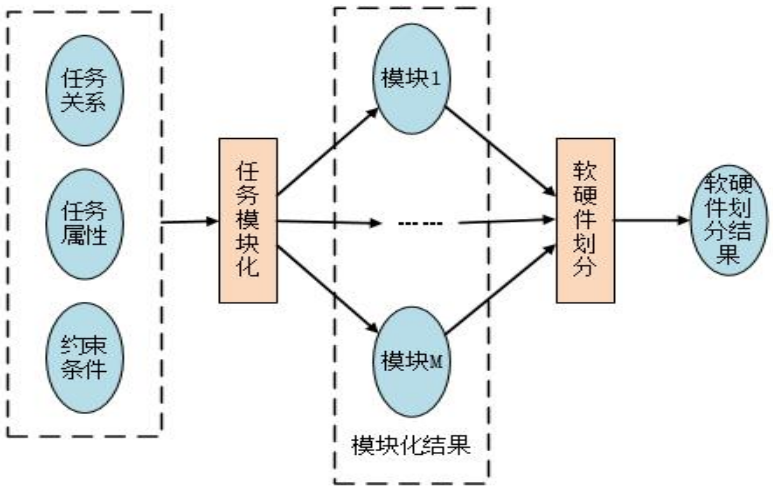

图 8-2 微系统划分过程示意图[40]

8.1.2. 基于多模块的微系统化示例

例 8.1 设一个系统有 8 个任务 A1，…，A8，他们间的通信代价和各任务的属性如表 8-1 和表 8-2 所示，现将这 8 个任务分成 3 个模块，使得每个模块不超过 3 个任务，使得通信消耗最小，每个模块依据任务的性能要求进行软硬件划分，使

得总体性能最优。

表 8-1 例 8.1 任务通信代价邻接矩阵

	A1	A2	A3	A4	A5	A6	A7	A8
A1		1	3	6	10	2	5	7
A2			8	2	10	12	13	15
A3				8	20	13	15	17
A4					19	18	17	5
A5						20	11	18
A6							12	13
A7								5
A8								

表 8-2 例 8.1 各任务的属性

任务	软时间	硬时间	硬面积	软可靠度	硬可靠度
A1	30	10	5	0.7	0.9
A2	40	12	6	0.65	0.8
A3	42	10	4	0.7	0.88
A4	35	9	5	0.76	0.91
A5	34	8	5	0.88	0.92
A6	22	4	2	0.89	0.92
A7	23	5	4	0.76	0.88
A8	20	6	5	0.75	0.86

解：（A）系统性能模块约束

第一步：划分模块。

依据（单链接）通信代价将 8 个任务划分成 3 个模块：M1：{A3，A5，A6}, M2：{A1，A4，A8}，M3：{A2，A7}。模块间通信代价：C_{M1M2}=108， C_{M1M3}=45，C_{M2M3}=68，整个通信代价为 221。

第二步：将每个模块进行软硬件划分，并计算其属性。

模块 M1：{A3，A5，A6}。 使用线性规划进行划分（在软时间<=150，硬件面积<=10 的约束下），模块的可靠度最高。划分结果为软件实现：A6；硬件实现：A3，A5；划分结果：软时间=22， 硬时间=18，硬面积=9。模块 M1 的可靠度=（A3 硬可靠度+A5 硬可靠度+A6 软可靠度）/3=2.69/3=0.897。如图 8-3(a)所示。

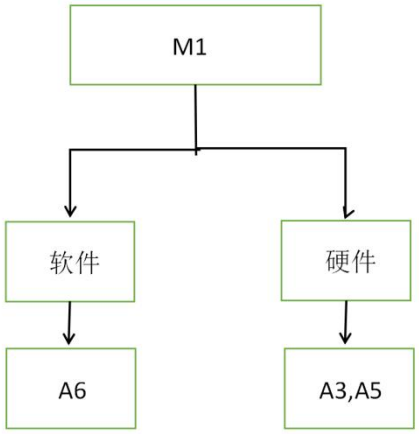

图 8-3(a) M1 划分结果图

模块 M2：{A1，A4，A8}。使用线性规划进行划分（软时间<=75，硬件面积<=10），模块的可靠度最高。划分结果为软件实现：A8；硬件实现：A1，A4；划分结果：软时间=20，硬时间=19，硬面积=10，模块 M2 的可靠度=0.853(2.56/3)。如图 8-3(b)所示。

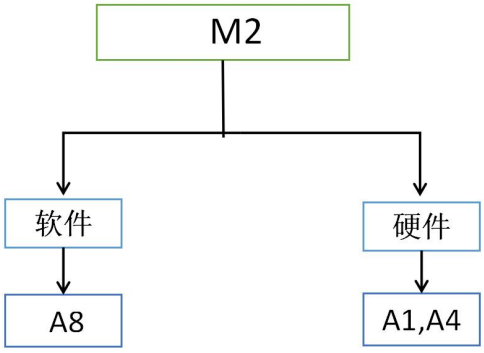

图 8-3(b) M2 划分结果图

模块 M3：{A2，A7}。使用线性规划进行划分，(软时间<=50，硬件面积<=9)，

模块可靠度最大，划分结果为软件实现：A7；硬件实现： A2; 划分结果：软时间=23， 硬时间=12， 硬面积=6，模块 M3 的可靠度=0.78(1.56/2)。如图 8-3(c)所示。

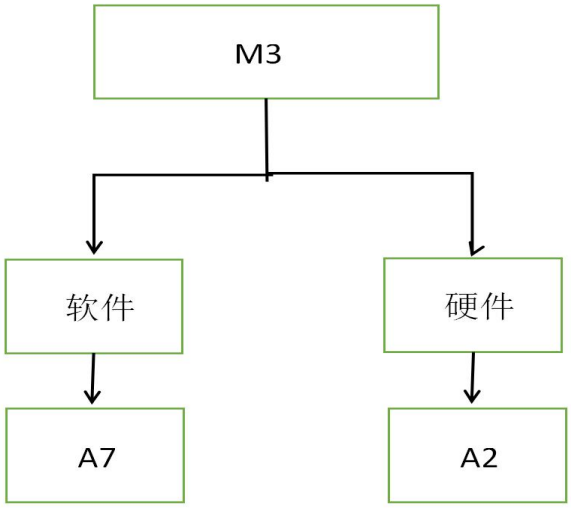

图 8-3(c) M3 划分结果图

三个模块属性统计结果如表 8-3 所示。

表 8-3 模块约束划分结果属性统计表

模块划分	软硬件划分	软时间	硬时间	硬面积	可靠度
M1： A3，A5，A6	软件：A6	22	18	9	0.897 (2.69/3)
	硬件：A3，A5				
M2： A1，A4，A8	软件：A8	20	19	10	0.853 (2.56/3)
	硬件：A1,，4				
M3： A2，A7	软件：A7	23	12	6	0.78 (1.56/2)
	硬件：A2				

第三步：计算整个系统的性能，见表 8-4。

表 8-4 划分后整个系统性能统计表

软时间	硬时间	硬面积	通信代价	可靠度

65	**49**	**25**	**221**	**0.843**

（B）系统性能整体约束

把系统性能进行整体考虑，就有这样的(整体)约束条件：如，软时间<=275，硬面积<=29,（3 个模块的约束条件之和）。依据这个约束条件，进行整体软硬件划分，就可以得到划分结果见表 8-5 属性统计表。由于任务的软硬件划分是整体进行的，没有考虑到模块是否一定有软实现和硬实现，因而就出现了模块 M2 和 M3 都是软件实现情况。这是不同于情况（A）的。

表 8-5 系统整体约束划分结果属性统计表

模块划分	软硬件划分	软时间	硬时间	硬面积	可靠度
M1： A3，A5，A6	软件：A5，A6 硬件：A3	56	10	4	0.883 (2.65/3)
M2： A1，A4，A8	软件： 硬件：A1，A4，A8	0	25	15	0.89 (2.67/3)
M3： A2，A7	软件: 硬件：A2，A7	0	17	10	0.84 (1.68/2)

例 8.2 使用 KL 算法将一个包含 8 个任务 T1，...，T8 的系统划分成 2 个模块，并对每个模块的任务划分成软件实现和硬件实现，使得：目标一：在一定系统成本约束下执行时间最短，设成本约束$C_{cons} = 1000$，硬件面积 AH_{cons}=19；目标二：在一定时间内系统成本最小，设总体执行时间约束$ET_{cons} = 80$，硬件面积约束 AH_{cons}=19。

表 8-6 例 8.2 任务通信代价邻接矩阵

	T1	T2	T3	T4	T5	T6	T7	T8
T1		1	3	6	10	2	5	7
T2			8	2	10	12	13	15

T3				8	20	13	15	17
T4					19	18	17	5
T5						20	11	18
T6							12	13
T7								5
T8								

表 8-6 例 8.2 任务属性表

任务 Ti	软件实现		硬件实现		
	软时间 TSi	软成本 CSi	硬时间 THi	硬成本 CHi	硬面积 AHi
T1	30	80	10	210	5
T2	40	100	12	263	6
T3	42	95	10	175	11
T4	35	76	9	182	7
T5	34	75	8	156	5
T6	22	63	7	163	2
T7	23	51	5	144	4
T8	20	49	6	99	8

解：分两步进行。

第一步。使用 KL 算法将任务进行模块化，划分结果为模块 M1：{T1，T3，T5，T8}、模块 M2：{T2，T4，T6，T7}，该划分结果的通信代价为：156。

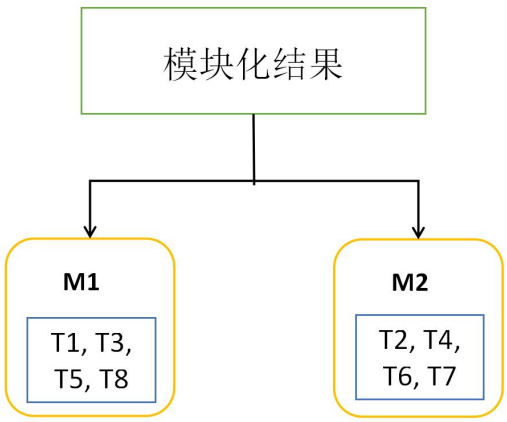

图 8-4 模块化结果图

第二步： 对每一个模块进行软硬件划分

完成了任务模块化之后，对每个模块内的任务进行软硬件划分，若需求为在一定系统开销约束下使得执行时间最短，设 Ccons = 1000; AHcons = 19 和 15 分别求解。使用 LINGO11 软件进行每个模块软硬件划分，得到结果如表 8-8 和 8-9 所示，统计结果见表 8-10。

表 8-8 模块 M1 软硬件划分结果

目标	约束条件	软实现	硬实现	时间	开销	面积
时间最短	Ccons<=1000 AHcons=19	T3	T1，T5，T8	66	560	18
	Ccons<=1000 AHcons=15	T3,T8	T1，T5	80	510	10

表 8-9 模块 M2 软硬件划分结果

目标	约束条件	软实现	硬实现	时间	开销	面积
时间最短	Ccons<=1000 AHcons=19		T2，T4，T6，T7	33	752	16
	Ccons<=1000 AHcons=15	T7	T2，T4，T6	51	659	15

表 8-10 系统划分结果

目标：时间最短	模块	任务	软任务	硬任务
开发代价 1000 硬件面积 19	M1	T1，T3，T5，T8	T3	T1，T5，T8
	M2	T2，T4，T6，T7		T2，T4，T6，T7
开发代价 1000 硬件面积 15	M1	T1，T3，T5，T8	T3,T8	T1，T5
	M2	T2，T4，T6，T7	T7	T2，T4，T6

第 8.2 节 面向多核的微系统化分

第 6 章介绍了多核划分算法，本节在第 6 章基础上，介绍面向多核的微系统化分算法。本节关键内容是任务划分。

任务划分是给任务分配计算单元、安排任务执行时间的过程，划分结果对软硬件协同设计中的系统性能有着关键性影响。

任务划分是智能嵌入式系统优化设计流程中的重要环节，暨考虑任务的执行顺序，又要考虑任务的软硬件配置，还要关注处理器间通信时延。本节扩展阅读文献可以选参考文献[40,41,42]。

图 8-6 是简化的多核系统异构片上系统结构图，Core1、Core2、...、CoreP 是为处理器，可以是如 CPU、MCU、GPU、ARM 等，而硬件区域可以是 ASIC 或 FPGA。处理器间、处理器与硬件区域通信通过总线实现，因而会产生时延。

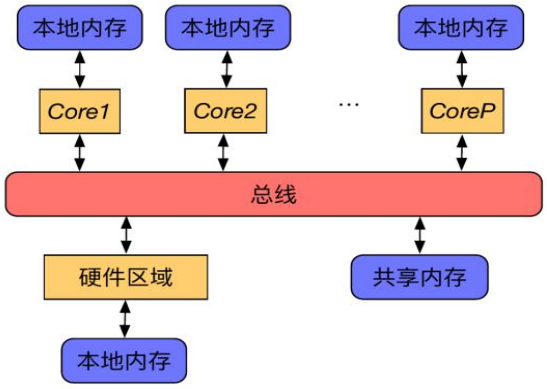

图 8-6 简化的多核系统异构片上系统结构[40]

含有 N 个任务图 G=(T，E)，T 是 N 个任务 Ti 之集，每个任务有 3 个性能属性 Ti=(TSi，THi，AHi)，其中 TSi 为软件执行时间、THi 为硬件执行时间、AHi 为硬件执行面积；E 为图 G 边的集合，(Ti，Tj)∈E 是一条有向边，e(Ti，Tj)为边(Ti，Tj)的权值，表示任务 Ti 到 Tj 的通信时间。调度后 Ti 得到 rTi=(pi, tsi, tfi)，其中 pi 表示任务 Ti 被分配到第 pi 处理单元，tsi 表示任务 Ti 的开始时间，tfi 表示任务 Ti 的结束时间。处理器之间处理器与硬件通信使用总线 BUS 完成，处理器内部通信以及硬件内部通信看成内部通信不在统计范围。

算法目标：Minimize $L = \max_{i\in\{1,2,\ldots,N\}} \{tfi\}$

约束条件：$S = \sum_{i=1}^{N} ai\ AHi$ <=L（当 Ti 为硬件任务是 ai=1，否则=0）。

注：使用任务的邻接矩阵表示任务间的通信时间。由于任务图是有向图，因此这个邻接矩阵不是对称阵。

以例 8.3 来介绍本节的方法。

例 8.3 已知一个系统有 11 个任务：T1，…，T11,每个任务有 3 个属性（软件执行时间，硬件执行时间，硬件执行面积），释放时间都是 0，任务属性图见图 8-7。系统的硬件执行面积约束条件为 S=18。将系统在 2 个处理器 Core1 和 Core2 和一块硬件板 H 上进行软硬件划分，在满足硬件约束条件前提下，系统执行时间最短，其中约定硬件板 H 可以 2 并行执行，即 2 个任务可以并行执行，系统通信通过 Bus 总线完成。

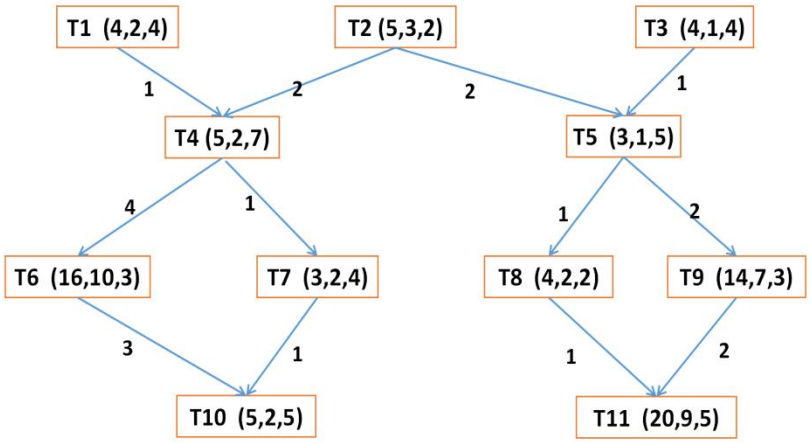

图 8-7 例 8.3 任务属性图[25]

8.2.1 基于硬件增益的软硬件划分

智能嵌入式系统的硬件面积是非常重要的一个约束指标，一方面硬件实现的任务执行时间比软件实现的任务执行时间要少得多，另一方面硬件实现的任务要占用硬件的面积。因此，从执行时间角度来说，尽可能多得安排任务由硬件实现，但硬件面积的约束导致在安排任务硬件实现时要考虑到硬件实现任务带来的增益，即若一个任务硬件实现的执行时间比软件实现的执行时间少得多，则硬件实现带来的增益要大。

定义 8.1 任务硬件实现增益 一个任务软件实现的执行时间减去硬件实现的执

行时间所得的差称为这个任务硬件实现的增益。

明显地，一个任务硬件现实增益越大，任务越应该使用硬件实现。因此，我们可以依据任务的软时间和硬时间数据来构造任务的硬件实现增益表。

例 8.3 中 11 个任务硬件实现增益表为：(T1, T2, T3, T4, T5, T6, T7, T8, T9, T10, T11)=（2, 2, 3, 3, 2, 6, 1, 2, 7, 3, 11）。从中可以得到任务 T11 的硬件实现增益最大（=11），而任务 T7 的硬件实现增益最小（=1）。因此任务 T11 最有可能硬件实现，而任务 T7 最没有可能硬件实现。

将任务图中的时间性能属性换成硬件实现增益，保留硬件执行面积，这样就得到一个新的任务图如图 8-8，使用第 6 章介绍的多核划分方法进行软硬件划分，但在计算任务优先级值时使用任务硬件实现增益值，在任务划分时累计硬件面积，若硬件面积超过了约束，则停止分配任务给硬件执行。

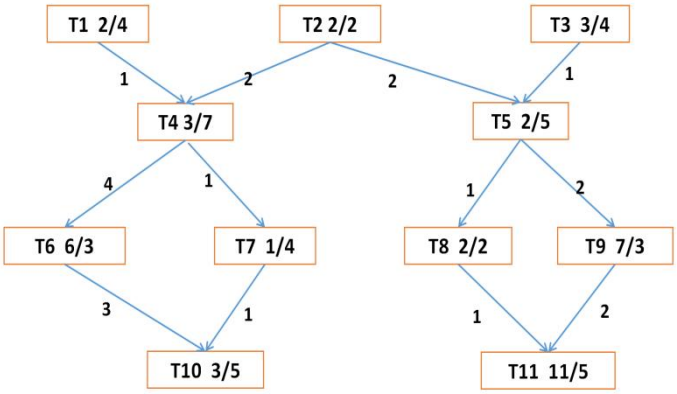

图 8-8 例 8.3 带有硬件实现增益的任务图

算法 8.2 基于硬件实现增益的软硬件划分算法 $HSPA_{HG}$

Input: 含有 n 个任务(t1，...，tn)的有向无环图，任务邻接矩阵，任务释放时间表，任务执行软时间表、硬时间表以及任务硬件面积表，k 个处理器 C1，...，Ck 和一块硬件 H，H 可以 2 并行执行，硬件面积约束条件<=L。

Output: n 个任务在 k 个处理器和硬件 H 上分配方案，每个任务起止时间，Bus 通信顺序表，包括任务通信指向以及任务通信起止时间，k 个处理器和硬件 H 的最大执行时间，每个处理器使用率。

第 1 步：计算 n 个任务的硬件实现增益；

第 2 步：依据 n 个任务的硬件实现增益，使用第 6 章任务优先级排序算法 TaPSA，建立任务的优先级表；
第 3 步：若任务优先级表为空则算法结束，否则按照任务优先级表级别高低检查其所有前驱任务以及指向该任务的所有通信是否执行完毕，在所有前驱和通信已执行完毕的任务中选择优先级别最高的任务，将该任务按照第 4 步进行分配。
第 4 步：若硬件空闲，并且硬件面积+该任务的硬件面积<=L 则把该任务分配给硬件执行同时更新已分配的硬件面积之和，按照该任务的硬件执行时间记录起止时间，从优先级表中删除该任务，转向第 7 步；若硬件不空闲或硬件空闲但硬件面积+该任务硬件面积>L 则转向第 5 步。
第 5 步：若处理器有空闲的，则把该任务分配给空闲较长的处理器，记录该任务的处理器编号并按照该任务的软件执行时间记录该任务起止时间，从优先级表中删除该任务，并转向第 7 步，否则转向第 6 步。
第 6 步：暂时不分配，停留在第 6 步，直到硬件空闲则转向第 4 步，或处理器空闲则转向第 5 步。
第 7 步：若任务执行完毕，则按照该任务的直接后继进行通信分配，记录该任务指向直接后继，按照通信时间记录通信的起止时间，直到所有直接后继都已通信分配。转向第 3 步。

例 8.4 按照算法 $HSPA_{HG}$ 对例 8.3 进行软硬件划分。

解： 按照算法 $HSPA_{HG}$ 分七步完成。

第 1 步：这 11 个任务硬件实现增益值分别为：2，2，3，3，2，6，1，2，7，3，11。

第 2 步：计算任务优先级以及优先级表。OPri(T11)=11，OPri(T10)=3，OPri(T9)=19，OPri(T8)=14，OPri(T7)=5，OPri(T6)=10，OPri(T5)=23，OPri(T4)=15，OPri(T3)=27，OPri(T2)=27，OPri(T1)=18。

优先级表：T2>T3>T5>T9>T1>T4>T8>T11>T6>T7>T10。

第 3-7 步系统划分过程见表 8-11：

表 8-11 第 3-7 调度过程表

时刻	分配前任务优先级表	任务分配	分配后优先级表	通信分配	硬面积累计
0	T2>T3>T5>T9>T1>T4>T8>T11>T6>T7>T10	rT3(H,0,1) rT2(H,0,3) rT1(C1,0,4) C2 空闲	T5>T9>T4>T8>T11>T6>T7>T10		s=4+2 =6<18
1	T5>T9>T4>T8>T11>T6>T7>T10	T3 完工 rT2(H,0,3) rT1(C1,0,4) C2 空闲		B(3->5,1,2)	
3	T5>T9>T4>T8>T11>T6	T2 完工		B(2->4,3,5),	

	>T7>T10	rT1(C1,0,4) rH/C2 空闲		B(2->5,3,5)	
4	T5>T9>T4>T8>T11>T6 >T7>T10	T1 完工 C1/C2/H 空闲		B(1->4,4,5)	
5	T5>T9>T4>T8>T11>T6 >T7>T10	rT5(H, 5,6), rT4(H ,5,7) C1/C2 空闲	T9>T8>T11>T6>T7 >T10		s=4+2+5+7 =18
6	T9>T8>T11>T6>T7>T10	T5 完工 T4(H ,5,7) C1/C2 空闲		B(5->8,6,7) B(5->9,6,8)	
7	T9>T8>T11>T6>T7>T10	T4 完工 rT8(C2,7,11) H/C1 空闲，	T9>T11>T6>T7 >T10	B(4->6,7,11) B(4->7, 7, 8)	
8	T9>T11>T6>T7>T10	rT8(C2,7,11) rT9(C1, 8,22) H 空闲	T11>T6>T7>T10		
11	T11>T6>T7>T10	T8 完工 rT9(C1, 8,22) rT6(C2,11,27) H 空闲	T11>T7>T10	B(8->11,11,12)	
22	T11>T7>T10	T9 完工 rT6(C2,11,27) rT7(C1,22, 25) H 空闲	T11>T10	B(9->11, 22,24)	
25	T11>T10	T7 完工 rT6(C2,11,27) rT11(C1,25,45) H 空闲	T10	B(7->10, 25, 26)	
27	T10	T6 完工 rT11(C1, 25,45) H/C2 空闲		B(6->10, 27,30)	
30	T10	rT10(C2,30,35) rT11(C1, 25,45) H 空闲	空		
35	空	T10 完工 rT11(C1, 25,45) C1/H 空闲			
45	空	T11 完工(结束) C1/C2/H 空闲			

划分结果统计：

C1：T1(0，4)，T9(8，22)，T7(22，25)，T11(25，45)，执行时间之和=4+14+3+20=41；

C2：T8(7，11)，T6(11，27)，T10(30，35)，执行时间之和=4+16+5=25；

H：T3(0，1)，T2(0，3)，T4(5，7)，T5(5，6)，执行时间之和=3+2=5，硬件执行面积之和=4+2+7+5=18。

整个系统完工时间为 45，硬件面积为 18， 处理器 C1 使用率=91.1%，处理器 C2 使用率=71.4%，硬件时间使用率=71.4%，硬件面积执行效率=100%.

表 8-12 依据算法 $HSPA_{HG}$ 划分结果分析表

处理器	C1	C2	H	硬件面积	BUS
分配方案	T1(0,4) T9(8,22) T7(22,25) T11(25,45) ，	T8(7,11) T6(11,27) T10(30,35)	T3(0,1)||T2(0,3) T4(5,7)||T5(5,6)	s=4+2+7+5	B(3->5,1,2), B(2->4,3,5) B(2->5,3,5),B(1->4,4,5) B(5->8,6,7)，B(5->9,6,8) B(4->6,7,11),B(4->7, 7,8) B(8->11,11,12),B(9->11,22,24) B(7->10, 25,26),B(6->10,27,30)
最大执行时间	45	35	7		
统计数据	31	25	5=（3+2）	18	
使用效率	68.9%	71.4%	71.4%	100%	

划分效果甘特图见图 8-9.

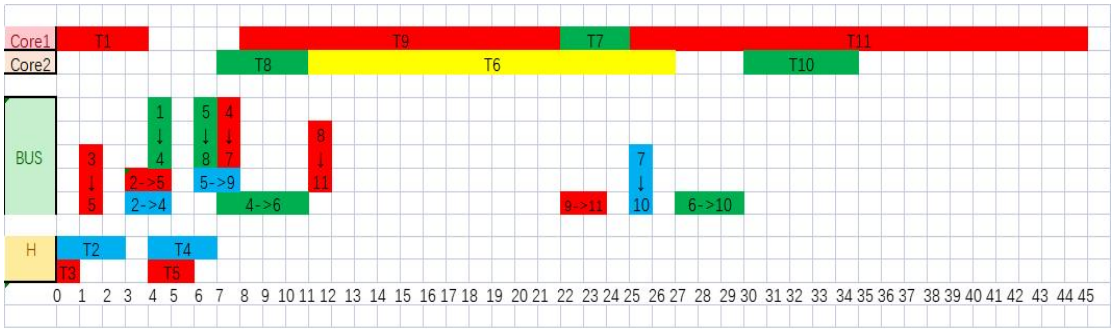

图 8-9 依据算法 $HSPA_{HG}$ 划分效果甘特图

8.2.2 基于硬件实现增益+硬件面积的软硬件划分 $HSPA_{HGA}$

从划分结果甘特图（图 8-9 可以看到，硬件实现的任务聚集在任务划分的开始阶段，这是因为 8.2.2 段介绍的方法是硬件优先原则，硬件执行面积仅在软硬件分配时验证是否满足硬件面积约束条件。

本段介绍在考虑硬件实现增益基础上增加硬件面积因素的软硬件划分算法，$HSPA_{HGA}$。该算法将任务按照硬件面积从小到大进行排序，得到按照硬件面积排

序的任务硬件优先执行表，硬件面积小优先执行级别高，这样安排是基于贪心策略---尽可能多地安排硬件执行任务。同时计算任务的平均面积，将硬件约束面积去除以任务平均面积，得到满足硬件面积约束的任务数。依据这个任务数和任务硬件优先执行表，确立应安排硬件执行的任务、应安排软件执行的任务以及不能确定的软件/硬件执行的任务。任务划分时将基于硬件实现增益的任务优先级表和任务硬件优先执行表结合在一起进行软硬件划分。

再以例 8.3 为例，以硬件面积为标准从小到大排序，得到任务硬件优先执行表为：T2(2)> T8(2)> T6(3)> T9(3)> T1(4)> T3(4)> T7(4)> T5(5)> T10(5)> T11(5) > T4(7)。硬件面积平均值为 4。硬件面积约束值为 18，18 除以 4=4.5，得到最多可以安排 5 个任务硬件执行。因此，T2，T8，T6，T9 最好硬件实现，T5，T10，T11，T4 最好软件实现，而 T1，T3，T7 中可能硬件实现，只要硬件面积满足约束条件。从这 11 个任务硬件实现增益值分别为：2(T1)，2(T2)，3(T3)，3(T4)，2(T5)，6(T6), 1(T7)， 2(T8)，7(T9)，3(T10)，11(T11)。

算法 8.3 基于硬件实现增益+硬件执行面积的软硬件划分算法 $HSPA_{HGA}$

Input: 含有 n 个任务(t1,...,tn)的有向无环图，任务邻接矩阵，任务释放时间表，任务执行软时间表、硬时间表以及任务硬件面积表，k 个处理器 C1,...,Ck 和一块硬件 H,H 可以 2 并行执行，硬件面积约束条件<=L。
Output: n 个任务在 k 个处理器和硬件 H 上分配方案，每个任务起止时间，Bus 通信顺序表，包括任务通信指向以及任务通信起止时间，k 个处理器和硬件 H 的最大执行时间，每个处理器执行效率。

第 1 步：计算 n 个任务的硬件实现增益，依据 n 个任务的硬件实现增益，第 6 章任务优先级排序算法 TaPSA，建立任务的优先级表；
第 2 步：依据 n 个任务的硬件执行面积，从小到大进行排序，面积相等时按照任务编号从小到大排序，建立任务硬件优先执行表。依据硬件面积约束值确立硬件实现的任务个数，再依据硬件优先执行表，确立可能硬件执行的任务序列和软件执行的任务序列。
第 3 步：若任务优先级表为空则算法结束，否则按照任务优先级表级别高低检查其所有前驱任务以及指向该任务的所有通信是否执行完毕，在所有通信已执行完毕的任务中选择优先级别最高的任务，若该任务是属于可能硬件执行的任务序列，则转向第 4 步；若该任务属于软件执行的任务序列，则转向第 5 步。
第 4 步：若累计硬件面积+该任务的硬件面积>L 则转向第 5 步，否则若硬件空闲，则把该任务分配给硬件执行同时更新已分配的硬件面积之和，按照该任务的硬件执行时间记录起止时间，从优先级表中删除该任务并转向第 7 步，若硬件不空闲则转向第 6 步。
第 5 步：若处理器有空闲的，若该任务在软件执行序列或是第 4 步转来的任务，则把该任务分配给空闲较长的处理器，记录该任务的处理器编号并按照该任务的软件执行时间记录该任

务起止时间，从优先级表中删除该任务，并转向第 7 步，若处理器不空闲则转向第 6 步。
第 6 步：暂时不分配，停留在第 6 步，直到硬件空闲则转向第 4 步，或处理器空闲则转向第 5 步。
第 7 步：若任务执行完毕，则按照该任务的直接后继进行通信分配，记录该任务指向直接后继，按照通信时间记录通信的起止时间，直到所有直接后继都已通信分配。转向第 3 步。

表 8-13 例 8.3 依据算法 $HSPA_{HGA}$ 划分过程表

时刻	分配前任务优先级表/硬件执行优先表	任务分配	分配后优先级表/分配后硬件执行优先表	通信分配	硬面积累计
0	T2>T3>T5>T9>T1>T4>T8 >T11>T6>T7>T10/ T2> T8> T6> T9> T1> T3 > T7	rT2(H,0,3) rT3(H,0,1) rT1(C1,0,4) C2 空闲	T5>T9>T4>T8>T11 >T6>T7>T10/T8 > T6> T9> T7		s=2+4 =6<18
1		T3 完工 rT2(H,0,3) rT1(C1,0,4) C2 空闲		B(3->5,1,2)	
3		T2 完工 rT1(C1,0,4) H/C2 空闲		B(2->4,3,5), B(2->5,3,5)	
4		T1 完工 C1/C2/H 空闲		B(1->4,4,5)	
5	T5>T9>T4>T8>T11>T6>T7>T 10/T8> T6> T9> T7	rT5(C2, 5,8), rT4(C1,5,10) H 空闲	T9>T8>T11>T6>T7 >T10 /T8> T6> T9> T7		
8		T5 完工 rT4(C1,5,10) H/C2 空闲		B(5->8,8,9) B(5->9,8,10)	
9	T9>T8>T11>T6>T7>T10 /T8> T6> T9> T7	rT4(C1,5,10) rT8(H,9,11) C2 空闲，	T9>T11>T6>T7 >T10 /T6> T9> T7		s=2+4+2 =8<18
10	T9>T11>T6>T7 >T10 /T6> T9> T7	T4 完工 rT8(H,9,11) rT9(H,10,17) C1/C2 空闲	T11>T6>T7>T10 /T6> T7	B(4->6,10,14) B(4->7, 10, 11)	s=2+4+2+3 =11<18
11	T11>T6>T7>T10 /T6 > T7	T8 完工 rT9(H,10,17) rT7(H,11,13) C1/C2 空闲	T11>T6>T10 /T6	B(8->11,11,12)	s=2+4+2+3 +4 =15<18
13		T7 完工 rT9(H,10,17) H/C1/C2 空闲		B(7->10, 13,14)	

14	T11>T6>T10 /T6	rT6(H,14,24) rT9(H,10,17) C1/C2 空闲	T11>T10		s=2+4+2+3 +4+3 =18=18
17	T11>T10	T9 完工 rT6(H,14,24) C2 空闲	T11>T10	B(9->11,17,19)	
19	T11 > T10	rT6(H,14,24) rT11(C2, 19,39) H/C1 空闲	T10		
24	T10	T6 完工 rT11(C2,19,39) C1/H 空闲	空	B(6->10,24,27)	
27		rT11(C2,19,39) rT10(C1,27,32) H 空闲			
32		T10 完工 rT11(C2,19,39) C1 /H 空闲			
39		T11 完工 C1 /C2/H 空闲			

划分结果分析见表 8-14。

表 8-14 例 8.3 依据算法 $HSPA_{HGA}$ 划分结果分析表

处理器	C1	C2	H	硬件面积	BUS
分配方案	T1(0,4) T4(5,10) T10(27,32)	T5(5,8) T11(19,39)	T2(0,3)|| T3(0,1) T8(9,11)||T9(10,17) T7(11,13)||T6(14,24)	s=2+4+2+3 +4+3	B(3->5,1,2), B(2->4,3,5) B(2->5,3,5),B(1->4,4,5) B(5->8,8,9)，B(5->9,8,10) B(4->6,10,14),B(4->7, 10, 11) B(8->11,11,12),B(7->10, 13,14) B(9->11,17,19),B(6->10,24,27)
最大执行时间	32	39	24		
统计数据	14	23	18=（3+15）	18	
使用效率	43.8%	59.0%	75%	100%	

注意的是： 硬件执行时间第一次硬件执行为 3， 而第二次执行时间从 9 到 24，因此为 15，两次合在一起硬件执行时间为 18。

划分结果甘特图见图 8-10.

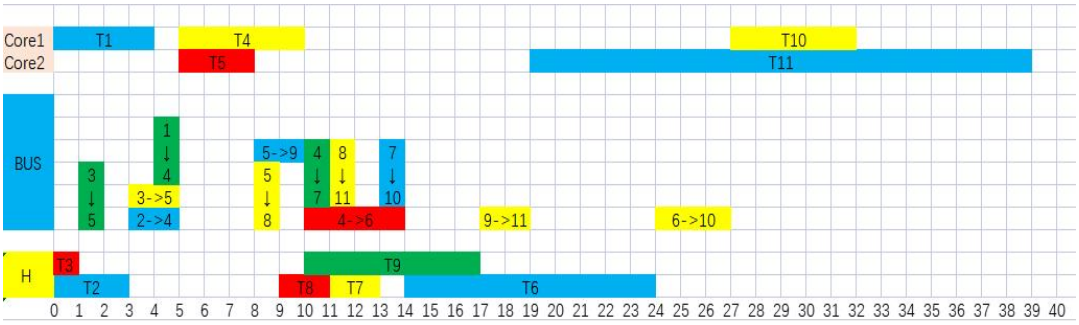

图 8-10 例 8.3 依据算法 $HSPA_{HGA}$ 划分效果甘特图

从划分效果来看，算法 $HSPA_{HGA}$ 与算法 $HSPA_{HG}$ 好些，硬件实现任务多些了，也比较分散些。但任务 T11 还是没有划分到硬件实现，这是因为 T11 任务的优先级别低，在划分过程中总是最后安排。

8.2.3 基于硬件面积+硬件实现增益的软硬件划分 $HSPA_{AHG}$

本段介绍的方法是首先把硬件实现的任务确定下来，这样软件实现的任务也就确定了下来，确定硬件实现任务时可以使用线性规划方法，求得硬件实现增益最大，而约束条件是硬件实现任务的面积小于等于硬件面积约束。在计算任务的优先级值时，若任务是硬件实现就计算硬件执行时间，若任务是软件实现就计算软件执行时间，同时把任务的出度和直接后继中的最大通信时间也都考虑在内。

任务优先级值计算公式：

$$O^{e}Pri(Ti)=\begin{cases}TSi+O(Ti)+\max_{Tj\in Suc(Ti)}\{e(Ti,Tj)+OPri(Tj)\}, & \text{若Ti软件实现}\\ THi+O(Ti)+\max_{Tj\in Suc(Ti)}\{e(Ti,Tj)+OPri(Tj)\}, & \text{若Ti硬件实件}\end{cases} \quad (公式8-1)$$

依据这个优先级值建立任务优先级表，依据优先级表进行软硬件划分，确定软硬件划分方案，统计方案的结果。

仍以例 8.3 为例，确定硬件实现任务。硬件实现增益从大到小排序如下：

T11(11，5)>T9(7，3)>T6(6，3)>T3(3，4)=T4(3，7)=T10(3，5)>T1(2，4)=T2(2，2)=T5(2，5)=T8(2，2)>T7(1，4)，其中括号里的第一个数字为硬件实现增益，第 2 个数字为硬件面积。

在硬件面积约束=18 的条件下，安排硬件实现任务使得硬件实现增益最大，使用第 5 章多目标划分方法可以求得，硬件实现任务有 T6，T8，T9，T10，T11，硬件实现增益之和为 59。相应地，软件实现任务为：T1，T2，T3，T4，T5，T7。注意到，先进行软硬件划分，任务 T11 划分给了硬件执行。

任务优先级值：

O^ePri(T11)=9， O^ePri(T10)=2， O^ePri(T9)=19， O^ePri(T8)=13， O^ePri(T7)=7，O^ePri(T6)=16，O^ePri(T5)=26，O^ePri(T4)=27，O^ePri(T3)=32，O^ePri(T2)=36，O^ePri(T1)=33。

建立优先级表：

T2>T1>T3>T4>T5>$\underline{T}$9>$\underline{T}$6>$\underline{T}$8>$\underline{T}$11>T7>$\underline{T}$10。($\underline{T}$i 表示任务 Ti 硬件实现)

算法 8.4 基于硬件面积+硬件实现增益的软硬件划分 $HSPA_{AHG}$

Input: 含有 n 个任务(t1，...，tn)的有向无环图，任务邻接矩阵，任务释放时间表，任务执行软时间表、硬时间表以及任务硬件面积表，k 个处理器 C1，...，Ck 和一块硬件 H，H 可以 2 并行执行，硬件面积约束条件<=L。

Output: n 个任务在 k 个处理器和硬件 H 上分配方案，每个任务起止时间，Bus 通信顺序表，包括任务通信指向以及任务通信起止时间，k 个处理器和硬件 H 的最大执行时间，每个处理器执行效率。

第 1 步：计算 n 个任务的硬件实现增益；

第 2 步：依据硬件面积约束条件和硬件实现增益最大化原则确定硬件实现任务和软件实现任务，获得软硬件划分结果；

第 3 步：依据 n 个任务的硬件实现增益，和软硬件划分结果，按照公式 8-1 计算任务实现的优先级值，并使用第 6 章任务优先级表方法确立确立任务优先级表；

第 4 步：若任务优先级表为空则算法结束，否则按照任务优先级表级别高低检查其所有前驱任务以及指向该任务的所有通信是否执行完毕，在所有通信已执行完毕的任务中选择优先级别最高的任务，若该任务是属于硬件执行的任务序列，则转向第 5 步；若该任务属于软件执行的任务序列，则转向第 6 步。

第 5 步：若硬件空闲，则把该任务分配给硬件执行同时更新已分配的硬件面积之和，按照该任务的硬件执行时间记录起止时间，从优先级表中删除该任务并转向第 8 步，否则转向第 7 步。

第 6 步：若处理器有空闲的则把该任务分配给空闲较长的处理器，记录该任务的处理器编号并按照该任务的软件执行时间记录该任务起止时间，从优先级表中删除该任务，并转向第 8 步，否则转向第 7 步。

第 7 步：暂时不分配，停留在第 7 步，直到硬件空闲则转向第 5 步，或处理器空闲则转向第 6 步。

第 8 步：若任务执行完毕，则按照该任务的直接后继进行通信分配，记录该任务指向直接后继，按照通信时间记录通信的起止时间，直到所有直接后继都已通信分配。转向第 4 步。

表 8-14 依据算法 $HSPA_{AHG}$ 对例 8.3 划分过程表

时刻	分配前任务优先级表	任务分配	分配后优先级表	通信分配
0	T2>T1>T3>T4>T5>$\underline{T}$9>$\underline{T}$6>$\underline{T}$8>$\underline{T}$11>T7>$\underline{T}$10	rT2(C1,0,5) rT1(C2,0,4) H 空闲	T3>T4>T5>$\underline{T}$9>$\underline{T}$6>$\underline{T}$8>$\underline{T}$11>T7>$\underline{T}$10	

4	T3>T4>T5>T̲9>T̲6>T̲8>T̲11>T7>T̲10	T1 完工 rT2(C1,0,5) rT3(C2,4,8) H 空闲	T4>T5>T̲9>T̲6>T̲8>T̲11>T7>T̲10	B(1->4,4,5)
5	T4>T5> T̲ 9> T̲ 6> T̲ 8> T̲ 11>T7>T̲10	T2 完工 rT3(C2,4,8) H/C1 空闲		B(2->4,5,7), B(2->5,5,7)
7	T4>T5> T̲ 9> T̲ 6> T̲ 8> T̲ 11>T7>T̲10	rT3(C2,4,8) rT4(C1,7,12) H 空闲	T5> T̲ 9> T̲ 6> T̲ 8> T̲ 11>T7>T̲10	
8		T3 完工 rT4(C1,7,12) H 空闲		B(3->5,8,9)
9	T5>T̲9>T̲6>T̲8>T̲11>T7>T̲10	rT4(C1, 7,12) rT5(C2 ,9,12) H 空闲	T̲9>T̲6>T̲8>T̲11>T7>T̲10	
12		T4 完工 T5 完工 H/C1/C2 空闲		B(5->8,12,13) B(5->9,12,14) B(4->6,12,16) B(4->7,12,13)
13	T̲9>T̲6>T̲8>T̲11>T7>T̲10	rT7(C1 13,16) rT8(H,13,15) C2 空闲	T̲9>T̲6>T̲11>T̲10	
14	T̲9>T̲6>T̲11>T̲10	rT7(C1 ,13,16) rT8(H,13,15) rT9(H,14,21) C2 空闲	T̲6>T̲11>T̲10	
15		T8 完工 rT7(C1 ,13,16) rT9(H,14,21) H/C1 空闲		B(8->11,15,16)
16	T̲6>T̲11>T̲10	T7 完工 rT6(H, 16,26) rT9(H,14,21) C1/C2 空闲	T̲11>T̲10	B(7->10,16,17)
21		T9 完工 rT6(H,16,26) H/C1/C2 空闲		B(9->11,21,23)
23	T̲11>T̲10	rT6(H,16,26) rT11(H,23,32) C1/C2 空闲	T̲10	
26		T6 完工 rT11(H, 23,32) H/C1/C2 空闲		B(6->10,26,29)

29	T10	rT10(H,29,31) rT11(H, 23,32) C1/C2 空闲	空	
31		T10 完工 rT11(H, 23,32) H/C1/C2 空闲	空	
32		T11 完工 H/C1/C2 空闲	空	

划分结果分析见表 8-16：

表 8-16 依据算法 $HSPA_{AHG}$ 划分结果分析表

处理器	C1	C2	H	硬件面积	BUS
分配方案	T2(0,5), T4(7,12) T7(13,16)	T1(0,4) T3(4,8) T5(9,12)	T8(13,15)||T9(14,21) T9(14,21)||T6(16,26) T6(16,26)||T11(23,32) T11(23,32)||T10(29,31)	s=2+3+3+ 5+5=18	B(1->4,4,5),B(2->4,5,7) B(2->5,5,7),B(3->5,8,9) B(5->8,12,13),B(5->9,12,14) B(4->6,12,16),B(4->7,12,13) B(8->11,15,16),B(7->10,16,17) B(9->11,21,23),B(6->10,26,29)
最大执行时间	16	12	32		29
统计数据	13	11	19	18	14
使用效率	81.3%	91.7%	59.4%	100%	48.3%

划分结果甘特图见图 8-11。

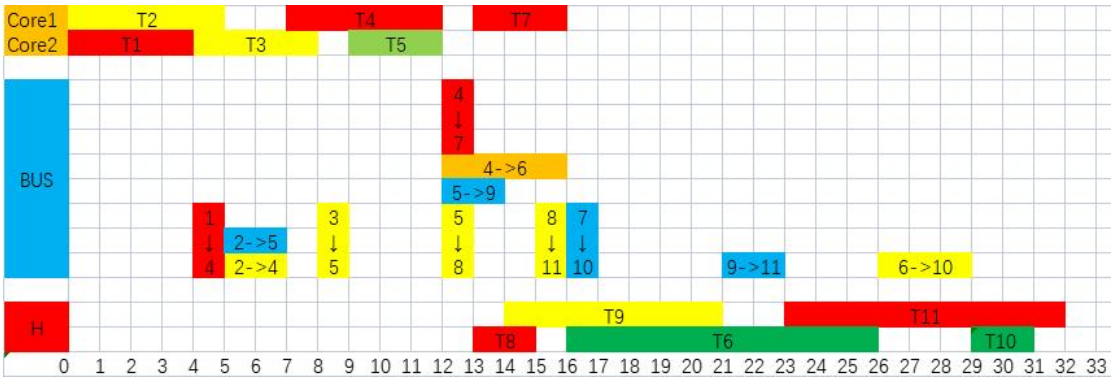

图 8-11 依据算法 $HSPA_{AHG}$ 例 8.3 划分效果甘特图

注：本段介绍了三种软硬件划分算法：基于硬件实现增益划分算法 $HSPA_{HG}$、基于硬件实现增益+硬件面积划分算法 $HSPA_{HGA}$ 和基于硬件面积+硬件实现增益划分算法 $HSPA_{AHG}$。它们的系统执行时间分别为 45、39 和 32，硬件实现任务个数分别是 4、6 和 5。硬件实现增益最大任务 T11 在 $HSPA_{AHG}$ 算法下才分配给硬件实现。

第 8.3 节 基于遗传算法的微系统划分算法 $HSPA_{GL}$

本节介绍文献[40]的微系统划分算法 **$HSPA_{GL}$**，该方法将遗传算法（Genetic Algorithm, GA）与任务表调度算法（List Scheduling with Static Priorities, LSSP）结合在一起，在确保满足多核片上系统硬件面积约束的前提下，能够同时给出软硬件划分结果与任务调度序列，并有效地缩短了系统总体运行时间。

本节介绍的方法与第 8.2 节还有不同之处是：BUS 线上不能同时进行两次数据传输，即只能有一个任务在 BUS 上进行数据传输，同时同一个处理器数据传输不在 BUS 线上进行，因而安排任务是尽可能地安排同一的处理器，即同核。

8.3.1 遗传算法与划分架构

遗传算法模拟一个人工种群的进化过程，通过选择(Selection)、交叉(Crossover) 以及变异(Mutation) 等机制，在每次迭代中保留一组优秀个体，种群重复以上步骤经过若干代演化后，理想情况下其适应度可以达到近似最优的状态。

遗传算法自被提出以来，得到了广泛的应用，特别是在函数优化、生产调度、模式识别、神经网络、自适应控制等领域发挥了巨大作用，提高了一些问题求解的效率。

遗传算法中，个体是基本单位，一个种群中拥有若干个体，每一次迭代中，种群中的个体数量不变。每个个体携带两个内容：染色体与适应度。

染色体表达了个体的某种特征；适应度函数值则用来评价一个个体的好坏，适应度函数值越大，个体越优秀，适应度函数是遗传算法进化的驱动力，也是进行自然选择的唯一标准，它的设计应结合求解问题本身的要求而定。

基于遗传算法的软硬件划分 $HSPA_{GL}$ 算法结合了遗传算法与任务调度算法，以遗传算法为主体，嵌入基于静态优先级的表调度算法，算法流程图如图 8-12 所示。

流程图中右边的部分为遗传算法部分。首先根据输入的任务图确定遗传算法中的种群大小、染色体长度、演化停止规则；然后进行种群初始化，计算初始种群中每个个体的适应度；随后不断进行个体选择、交叉、变异等操作；直至满足设定的演化停止条件。

流程图左边部分为任务调度部分。在遗传算法为每个个体计算适应度时都

被调用一次，并将得到的调度结果传回给遗传算法。该部分采用了基于静态优先级的表调度算法。对于每一代种群中的每一个个体，其染色体序列表示了一种对任务集合的软硬件划分。在进行调度时，首先根据任务图以及其染色体编码计算每个任务的优先级；然后初始化就绪任务队列，不断选择队列中优先级最高的任务，为其分配最佳处理器并计算其开始和结束时间，直到所有任务完成调度；最终得到该个体的调度结果。

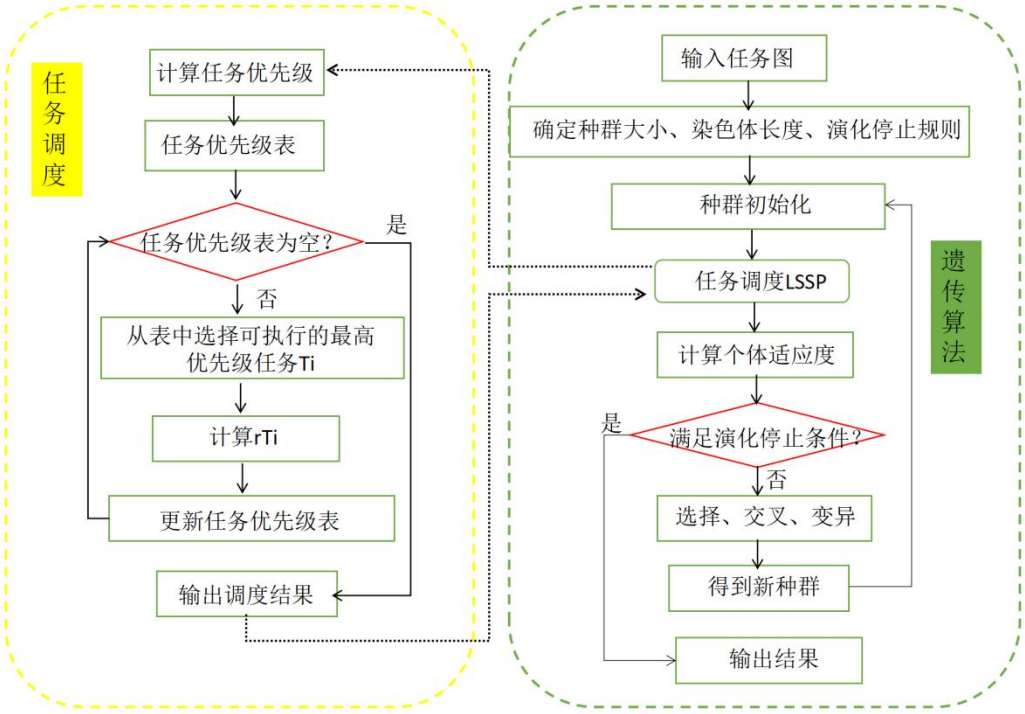

图 8-12 HSP_{AL} 算法流程图

8.3.2 遗传算法相关参数

本算法中遗传算法部分相关参数设定如下：

a. **种群大小**：设置种群大小Size = 2N，种群过小会降低基因多样性，过大则会影响算法执行效率。

b. **染色体长度**：N（=任务个数），一个个体（包含所有的任务）对应一种染色体编码，一个染色体有 N 个基因， 第 i 个基因对应任务 Ti。

c. **编码**：采用二进制编码，染色体中每一个基因，若为 0 表示该任务用软件实现，1 则表示该任务用硬件实现。

如：00010100101（软件任务：T1，T2，T3，T5，T7，T8，T10；硬件任务：T4，T6，T9，T11）

d. 演化停止规则：设置种群演化到 Gen = 3N 代，代数过少可能找不到最佳个体，过多则会导致算法执行时间过长。

e. 初始化种群：为保证本算法能找到满足硬件面积约束的解，使用贪心思想生成一个个体 α 加入到初始种群。其他 2N-1 个个体则随机产生。若找不到个体 α 则说明该问题在当前硬件面积约束下无解。

初始个体的生成算法 GeneAlpha 描述如算法 8.5，分成两个步骤。

步骤 1：计算每个任务 Ti 的硬件实现增益：Bi=TSi-THi，并按照硬件实现增益进行从大到小进行任务排序。

步骤 2：迭代查找最大硬件增益任务，将其放到硬件执行集合中，直到不满满足面积约束为止。将硬件集合中的任务对应的基因设为 1，其余设为 0，则得到了一个初始个体 α 。

算法 8.5 初始个体α生成算法 GeneAlpha

```
Input: Task Graph G;  The set T of Tasks of G; Hardware area constraint SL;
Output:  Chromosome for individual α;
1:  bool α[N] =0;
2:  for i = 1 to N do
3:    Bi = TSi -THi;
3:  end for
4:  Sort tasks according to hardware benefit;
6:  Initialize S = 0; THW = ∅ ;
7:   while S < SL do
8:      Select Ti with highest hardware benefit Bi;
9:      if S + AHi <=SL then
10:        THW = THW ∪ {Ti};
11:        S = S + AHi;
12:     end if
13:     T = T\ {Ti};
14:  end while
15:  for i = 1 to N do
16:     if Ti ∈ THW then α[i] = 1 else α[i] = 0;
18:     end if
19:  end for
20:  return α
```

f. 选择机制(Selection)：采取精英机制与轮盘赌方法相结合的方法。

精英机制的意义在于保留种群中的优秀个体，将每一代种群中个体适应度按从大到小排序，保留前 1/4 的个体直接进入下一代。下一代中其他个体由使用

轮盘赌方法选出的父母进行交叉产生。

轮盘赌方法选出下一代父母：随机地给每个任务分配概率，适应度越高的任务指定的概率越大。然后对一个任务随机产生一个概率，若大于原随机分配概率则这个任务被选中作为下一代的父母。共选出剩下 3/4 个父母。

g. 交叉操作(Crossover)产生下一代：采用两点交叉法，选择出父母个体后，随机选取其染色体中两点间的基因进行交换以产生下一代个体。

如：以两个染色体长度为 11 的个体为例，交换其 5-7 位的基因，产生两个新个体的过程如图 8-13 所示。

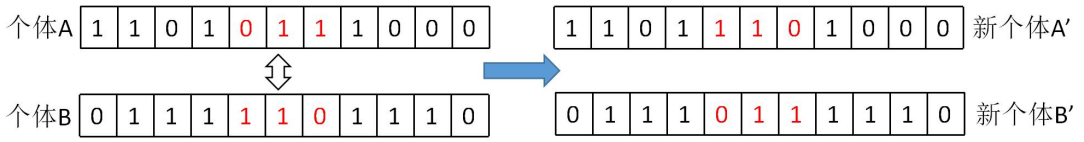

图 8-13 染色体交叉示例

h. 变异操作(Mutation)：对新种群中的每个个体，按照 Pm =0.5%的变异概率进行变异操作，每次变异可随机改变 1-3 个基因，基因 0 变异为基因 1，而基因 1 变异成基因 0。

如：某个体第 4/5 位基因发生突变的结果如图 8-14 所示。

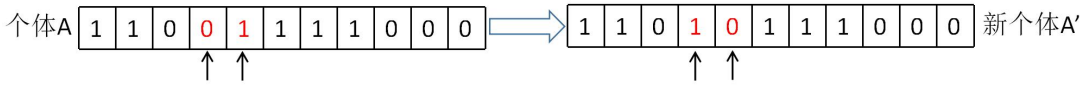

图 8-14 染色体变异示例

i. 适应度函数(Fitness)：对于每个个体，其染色体序列表示了一种对任务的软硬件划分方案，该划分中硬件任务面积之和 S，对划分后的任务采用基于静态优先级的调度算法得到调度总时长 L。算法目标为调度总时长最短，因此要求调度总时长越短的个体适应度越高。所需硬件面积超过了面积约束的个体是无效个体，引入惩罚项已降低其适应度。根据以上分析，设计出如下的个体适应度函数为：

$$Fitness = \frac{1}{\omega_1 \cdot e^{\frac{S-S_L}{S_L}} \cdot \frac{|S-S_L|}{S_L} + \omega_2 \cdot \frac{L}{\sigma}}$$

其中$e^{\frac{S-S_L}{S_L}} \cdot \frac{|S-S_L|}{S_L}$是惩罚项，当个体为无效个个体时，超过硬件面积约束越多受到的惩罚就越大，且为了提高硬件利用率，对占用硬件面积过小的个体也进行一定

的惩罚。$\sigma = \sum_{i=1}^{N} TSi$ 是归一化因子，目的是为了减少硬件面积和执行时间在数量级上带来的差异。ω_1 和 ω_2 为惩罚项与调度长度的权重，设 $\omega_1 = 0.7$，$\omega_2 = 0.3$ 以尽量降低无效个体的适应度。

8.3.4 表调度算法

本算法中任务调度采用了基于静态优先级的表调度算法 LSSP(List Scheduling with Static Priorities)。该算法的思想是首先计算出每个任务的的静态优先级；然后按照优先级由高到低对任务进行排序，构建一个任务队列；从此队列中取出具有最高优先级的任务，将其分配到当前最优的处理器上。

使用静态优先级 LSSP（算法 4）具体过程如下：

步骤 1：按照公式 8-2 计算任务静态优先级值。定义任务 Ti 的优先级值 pri(Ti) 为：

$$pri(Ti) = \begin{cases} TSi + \max_{Tj \in Suc(Ti)} \{e(Ti,Tj) + pri(Tj)\}, & \text{若Ti软件实现} \\ THi + \max_{Tj \in Suc(Ti)} \{e(Ti,Tj) + pri(Tj)\}, & \text{若Ti硬件实现} \end{cases} \quad (公式8-2)$$

在计算时，为了降低算法复杂度，避免递归调用重复计算，需先将任务按照任务图进行拓扑排序，按照拓扑序从后向前计算任务的优先级，即可保证在原 DAG 图中自底（叶节点）向上（根节点）计算。

步骤 2：初始化就绪任务队列，任务就绪是指该任务的所有前驱任务已经执行完成。初始任务序列中包含直接前驱任务集为空集的所有任务。

步骤 3：当就绪任务队列非空时，从就绪任务队列中选出当前优先级最高的任务 Ti，计算 rTi=（pi，tsi，tfi)。若该任务为硬件任务则分配至硬件执行，若为软件任务则为其选择最佳处理器核；其中最佳处理器核的选择标准为选择能使任务最早启动的核（若在几个核上任务的最早启动时间相同，那么选取能节省通信代价的核（优先安排与其直接前驱任务在同一个核中），若结果仍不唯一则选取序号最小的处理器核），即

$$pi = \begin{cases} j, \text{ j使得} start(Ti, j) = \min_{k \in \{1,2,\ldots,P\}} \{start(Ti,k)\}, & \text{若Ti软件实现} \\ 0, & \text{若Ti硬件实件} \end{cases}$$

任务之间通信在 BUS 上只能允许一个数据在传输，这点是不同于 8.2 节的任务通信方式。

算法 8.6 任务调度算法 LSSP

Input: Task graph G; The set T of Tasks of G; Number of processors P;
Chromosome (Hardware/Software Partitioning Result)；
Output: Scheduled task set r(T);
1: Tready =∅ ;

```
2: r(T) =∅ ;
3: Save the sequence numbers of tasks in list numList by topological order;
4: for i = N to 1 do
5:     num = numList.get(i);
6:     Compute pri(Tnum) according to the formula 8-2;
7:     if Pre(Tnum) =∅  then
8:         Tready = Tready ∪ {Tnum};
9:     end if
10: end for
11: while |r(T)| ! = |T| do
12:     Select Ti with highest priority from Tready;
13:     Compute r(Ti) = (pi, tsi, tfi);
14:     r(T) = r(T) ∪ {rTi};
15:     Tready = Tready \ {Ti};
16:     for Tj ∈ Suc(Ti) do
17:         if Tj is ready then
18:             Tready = Tready ∪{Tj};
19:         end if
20:     end for
21: end while
22: return r(T);
```

其中 start(Ti, j)表示任务 Ti 被分配到了第 j 个计算单元时的开始时间。每个任务需得到其所有直接前驱任务传递至它的数据才可开始执行，数据的传递只有当 BUS 不被其他数据传输占用时才可以进行，则

$$\mathrm{start(Ti, j) = Max\{ej, max_{\{k|Tk\in Pre(Ti)\}}\{tfk + e(Ti, Tk) + tocc\}\}}$$

其中 ej 表示 Corej 的最早空闲时间，也就是当前已经分配到该处理器核上的任务中最晚的结束时间；tocc 表示数据需要传输时 BUS 被其他任务间的数据传输所占用时间。

任务的开始时间 tsi 为任务最早启动时间，即：

$$\mathrm{tsi = min_{j\in\{1,2,...,P\}}\{start(Ti, j)\}}$$

由于任务一旦开始不会被打断，所以任务结束时间为任务的开始时间加上在相应处理单元上的执行时间，即

$$\mathrm{tfi} = \begin{cases} \mathrm{tsi+TSi,} & \text{若Ti软件实现} \\ \mathrm{tsi+THi,} & \text{若Ti硬件实件} \end{cases}$$

当一个任务完成之后，逐一检查其后续任务是否就绪，若有新的就绪任务就

将其加到就绪任务队列中进行调度，不断重复此过程，直到所有任务调度完毕，此时可得出调度总时长 L 和占用硬件面积和 S。

定理 8.1 LSSP 算法的时间复杂度为 $O(PN^2)$。

证明：类似于第 6 章定理 6.4 的证明。

8.3.4 遗传微系统划分算法 $HSPA_{GL}$

在遗传算法达到停止演化条件时，选择当前种群中适应度最高的个体作为本次 $HSPA_{GL}$ 算法运行的最优个体。最优个体调度中最后一个任务完成的时间则为系统总体运行时间 L，该个体染色体编码为 1 的基因所对应的任务的硬件面积和为系统使用的硬件面积 S。

我们还可以计算得到系统中总的通信代价为 $\sum_{(Ti,Tj)\in E} C(Ti, Tj)$，其中

$$C(Ti, Tj) = \begin{cases} 0, & \text{若Ti和Tj被分配到相同的处理单元} \\ e(Ti, Tj), & \text{若Ti和Tj被分配到不同的处理单元} \end{cases}$$

算法 8.7 基于遗传算法的软硬件划分算法 $HSPA_{GL}$

Input: Task graph G; Number of processors P; Probability of mutation Pm;
Output: Result of Hardware/Software Partitioning and Scheduling;
1: *Size* = 2N; *Gen* = 3N; *gen* = 0;
2: Generate individual α by using GeneAlpha;
3: Generate other individuals of the first generation *pop* randomly;
4: **for** every individual in *pop* **do**
5: r(T)---Hardware/Software Partitioning and Scheduling by LSSP;
6: Compute the fifitness of r(T) by using **Fitness**;
7: **end for**
8: **while** *gen* < Gen do
9: newpop =∅ ;
10: **Select** the individuals with highest fifitnesses and add them to *newpop*;
11: **while** |*newpop*|< *Size* **do**
12: Pick up two individuals *indi1*, *indi2*;
13: (*indi3, indi4*) = **cross**(*indi1, indi2*);
14: Add *indi3* and *indi4* to *newpop*;
15: **end while**
16: **for** every individual in *newpop* do
17: **if** random (0, 1) < Pm then
18: **Mutation**;
19: **end if**
20: **end for**
21: *pop* = *newpop*; *gen + +*;
22: **for** every individual in *pop* do
23: r(T)---Hardware/Software Partitioning and Scheduling by LSSP;

24: Compute the fifitness of r(T) by using **Fitness**;
25: **end for**
26: **end while**
27: Pick up the individual with highest fifitness;
28: **return** Result for Hardware/Software Partitioning and Scheduling;

定理 8.2 $HSPA_{GL}$ **算法复杂度是** $O(PN^4)$**。**

证：在 $HSPA_{GL}$ 算法过程中，种群大小、演化代数等参数在 O(1)时间内即可完成设置。随后进行初始种群的生成：首先要生成个体 α，对每个任务计算硬件收益需要 O(N)时间，随后对硬件收益排序需要 $O(N^2)$时间，接下来将任务加入到硬件集合中最多需要 O(N)时间，因此 GeneAlpha 的复杂度为 $O(N^2)$。 之后随机生成初始化种群中的其他个体，每个个体的生成需要 O(N)时间，共需 $O(N^2)$时间。则完成整个种群初始化过程需要 $O(N^2)$时间。

接下来进入演化过程，对每一代种群来说，有 2N 个个体需要调用 LSSP 算法，其复杂度为 $O(PN^3)$；为所有个体计算适应度复杂度为 O(N)，对个体按照适应度排序时间复杂度为 $O(N^2)$，选择、交叉、变异过程复杂度均为 O(N)，因此每一代操作的复杂度为 $O(PN^3)$。而算法共演化 3N 次，因此需要 $O(PN^4)$ 时间。演化结束后在 O(N) 时间内可以找到最优个体。综上所述，$HSPA_{AL}$ 算法的时间复杂度为 $O(PN^4)$。

8.3.5 遗传微系统划分算法 $HSPA_{GL}$ 示例

本段使用 $HSPA_{GL}$ 算法对 8-7 所示任务图进行划分与调度，设定处理器内核数 P=2，硬件面积约束 S_L=18。

首先根据任务数 N =11,设置种群大小为 22，染色体长度为 11，演化代数为 33 代。

然后初始化种群，根据 GeneAlpha 算法计算得到个体 α 。

步骤 1：计算得到每个任务硬件收益时间 B1=2, B2=2, B3=3, B4=3, B5=2, B6=6, B7=1, B8=2, B9=7, B10=3, B11=11。

步骤 2：依次选出硬件收益最大的任务 T11, T9, T6, T4，这四个任务硬件面积之和 S=5+3+3+7=18。得到的初始个体 α 为 00010100101。

其余 21 个初始个体随机产生，共产生 22 个个体，为初始种群。

随后对种群中的每个个体使用 LSSP 算法进行调度并计算调度结果个体的适应度。以个体 α 为例，使用 LSSP 算法对其进行调度过程如下。

步骤 1：计算每个任务的静态优先级，按照任务的拓扑序倒序，自底向上依次计算：

pri(T10)=5，pri(T11)=9，pri(T6)=18，pri(T7)=9，pri(T8)=14，pri(T9)=18，pri(T4)=24，pri(T5)=23，pri(T1)=29，pri(T2)=31，pri(T3)=28。

排出优先级表：T2>T1>T3>T̲4>T5>T̲6>T̲9>T8>T7>T̲11>T10，其中T̲4，T̲6，T̲9，T̲11 是硬件实现。

步骤 2：初始优先级表：T2>T1>T3>T̲4>T5>T̲6>T̲9>T8>T7>T̲11>T10。

步骤 3：不断从非空的优先级表中挑出可以调度的优先级最高的任务，将硬件任务分配至硬件，将软件任务分配最佳处理器核。

首先，优先级表中优先级最高的是 T2，Core1 与 Core2 都可以使该任务在 0 时刻开始，选择为其分配 Core1，开始时间为 0，结束时间为 4。然后选择 T1，此时 Core1 被 T2 占用，为了使其尽快开始，将 T1 分配到 Core2。同样，对 T3，将其分配到 Core2，这样它可以在时刻 4 开始。到了时刻 5，T1 和 T2 执行结束，T4 的所有前驱任务已经结束，T4 是可以调度的优先级最高的任务。T4 是硬件任务，所以直接将其分配至硬件，但是在时刻 7 之前 T4 不能开始，因为它需要等待从 T1 和 T2 通过 BUS 总线传递给它的数据。

类似地，执行此过程直到所有任务完成调度，调度过程详见表 8-15。

表 8-15 个体 α 调度过程表

时刻	分配前任务优先级表	任务分配	分配后优先级表	通信分配
0	T2>T1>T3>T̲4>T5>T̲6> T̲9>T8>T7>T̲11>T10	rT2(1,0,5) rT1(2,0,4) H 空闲	T3>T̲4>T5>T̲6> T̲9>T8>T7>T̲11>T10	
4	T3>T̲4>T5>T̲6> T̲9>T8>T7>T̲11>T10	T1 完工 rT2(1,0,5) rT3(2,4,8) H 空闲	T̲4>T5>T̲6>T̲9>T8> T7>T̲11>T10	B(1->4,4,5)
5		T2 完工 rT3(2,4,8) H 空闲		B(2->4,5,7)
7	T̲4>T5>T̲6>T̲9>T8> T7>T̲11>T10	rT3(2,4,8) rT4(0,7,9)	T5>T̲6>T̲9>T8>T7> T̲11>T10	

8		T3 完工 rT4(0,7,9)		B(3->5,8,9)
9	T5>T̲6>T̲9>T8>T7> T̲11>T10	T4 完工 rT5(1 ,9,12) rT6(0,9,19)	T̲9>T8>T7>T̲11>T10	B(4->7,9,10) T4 与 T6 都是硬件实现，因而不需要传输数据
10	T̲9>T8>T7>T̲11>T10	rT5(1 ,9,12) rT6(0,9,19) rT7(2,10,13)	T̲9>T8>T̲11>T10	
12	T̲9>T8>T̲11>T10	T5 完工 rT8(1,12,16) rT6(0,9,19) rT7(2,10,13)	T̲9>T̲11>T10	B(5->9,12,14) T8 安排同 T5 在一个处理器 1，因而不需要传输数据
13		T7 完工 rT8(1,12,16) rT6(0,9,19)	T̲9>T̲11>T10	B(7->10,13,14)
14	T̲9>T̲11>T10	rT8(1,12,16) rT6(0,9,19) rT9(0,14,21)	T̲11>T10	
16		T8 完工 rT6(0,9,19) rT9(0,14,21)	T̲11>T10	B(8->11,16,17)
19		T6 完工 rT9(0,14,21)	T̲11>T10	B(6->10,19,22)
21	T̲11>T10	T9 完工 rT11(0,21,30)	T10	T9 和 T11 都是硬件实现，因而不需要传递数据。
22	T10	rT11(0,21,30) rT10(2,22,27)	空	B(6->10,26,29)
27		T10 完工		
30		T11 完工		

对个体 α 调度得到的结果如图 8-16 所示，调度长度 L 为 30。计算得到个体 α 适应度为

$$Fitness = \frac{1}{0.7 \cdot e^{\frac{18-18}{18}} \cdot \frac{|18-18|}{18} + 0.3 \cdot \frac{30}{83}} = 9.222$$

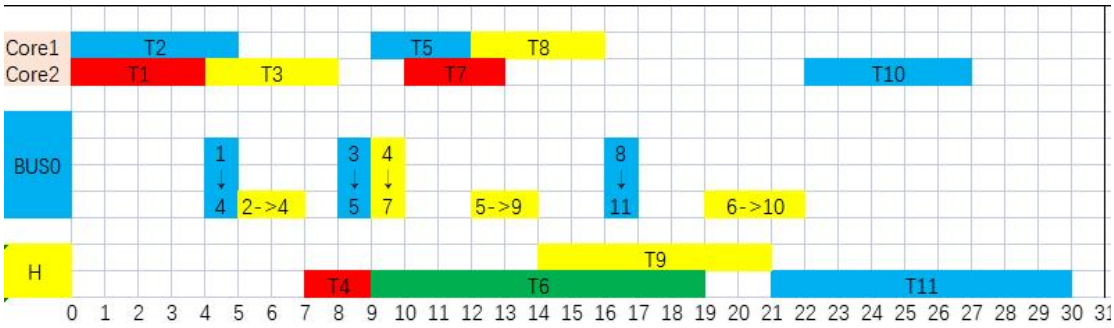

图 8-16 个体 α 调度结果

以个体 α 为初始种群的种子，使用 $HSPA_{GL}$ 算法，对初始种群中每个个体进行调度，得到结果后计算其个体适应度，然后按适应度大小进行排序；选出每一代最优的个体，再进行选择、交叉、变异等一系列遗传算法操作，直到演化至 33 代，找到最优个体，记录其调度结果。最终选出最优的个体为 β =01000100111，由此可以得到软硬件划分结果：硬件任务计划 THW={T2, T6, T9, T10, T11}，硬件面积之和等于 18。β 调度过程见表 8-6。

表 8-6 个体 β 调度过程

时刻	分配前任务优先级表	任务分配	分配后优先级表	通信分配
0	T1>T2>T3>T4>T5>T6>T9>T8>T7>T11>T10	rT1(1,0,4) rT2(0,0,3) rT3(2,0,4)	T4>T5>T6>T9>T8>T7>T11>T10	
3		T2 完工		B(2->4,3,5)
4		T1 与 T3 完工		
5	T4>T5>T6>T9>T8>T7>T11>T10	rT4(1,5,10)	T5>T6>T9>T8>T7>T11>T10	B(2->5,5,7) 不能同时通信
7	T5>T6>T9>T8>T7>T11>T10	rT5(2,7,10)	T6>T9>T8>T7>T11>T10	
10	T6>T9>T8>T7>T11>T10	T4 与 T5 完工 rT7(2,10,13) rT8(1,10,14)	T6>T9>T11>T10	B(5->9,10,12)
12	T6>T9>T11>T10	rT9(0 ,12,19)	T6>T11>T10	B(4->6,12,16)
13		T7 完工	T6>T11>T10	
14		T8 完工	T6>T11>T10	
16	T6>T11>T10	rT6(0,16,26)	T11>T10	B(7->10,16,17)
17				B(8->11,17,18)

19	T11>T10	T9 完工 rT11(0,19,28)	T10	T9 与 T11 同核
26	T10	T6 完工 rT10(0,26,28)		T6 与 T10 同核
28		T10 与 T11 完工		

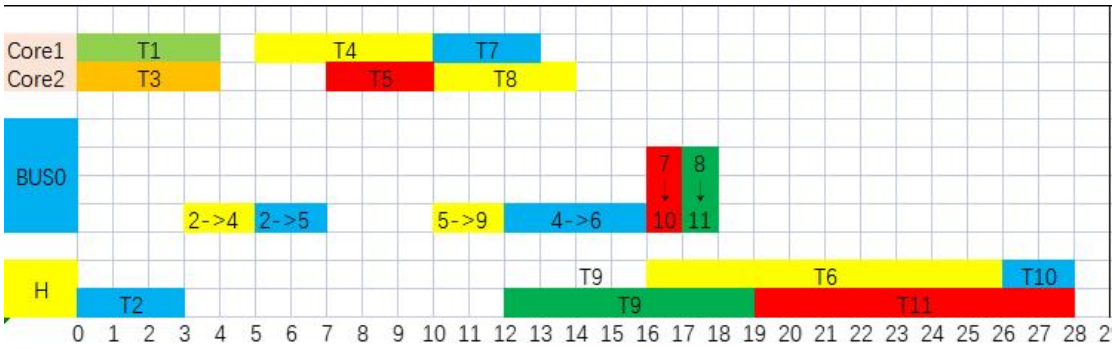

图 8-17 个体 β 调度结果

表 8. 个体 β 调度结果

任务 Ti	调度后任务 rTi			
	处理单元 pi	开始时间 tsi	结束时间 tfi	BUS 通信
T1	1（Core1）	0	4	
T2	0(硬件)	0	3	B(2->4, 3, 5) B(2->5, 5, 7)
T3	2(Core2)	0	4	
T4	1(Core1)	5	10	B(4->6, 12, 16)
T5	2(core2)	7	10	B(5->9, 10, 12)
T6	0(硬件)	16	26	
T7	1(Core1)	10	13	B(7->10, 16, 17)
T8	2(Core2)	10	14	B(8->11, 17, 18)
T9	0(硬件)	12	19	
T10	0(硬件)	26	28	
T11	0(硬件)	19	28	

调度算法 $HSPA_{GL}$ 产生的个体 β 的调度长度 L 为 28。计算得到个体 β 适应度为

$$Fitness = \frac{1}{0.7 \cdot e^{\frac{18-18}{18}} \cdot \frac{|18-18|}{18} + 0.3 \cdot \frac{28}{83}} = 9.8814$$

比 α 的适应度提升了 7.15%。

注：若在调度时增大 BUS 总线的性能，允许两个任务同时通信传递数据，则对个体 β 调度长度 L 可缩短到 26，此时个体 β 的适应度为 10.641，比 α 的适应度提升了 15.4%。

第 8.4 节 本章小结

本章介绍了微系统划分方法，包括基于模块的微系统划分划分方法、基于多核的微系统划分方法，以及基于遗传算法的微系统划分算法，从不同角度讨论智能嵌入式系统的软硬件划分。若不太重视通信延迟问题，则考虑使用基于模块的微系统划分方法，若很在意通信延迟则考虑使用基于多核的微系统划分方法、以及基于遗传算法的微系统划分算法。本章使用的例子还是比较简单，对于复杂情形需要使用其他方法进行微系统划分。如把遗传算法应用于微系统划分可参考文献[40]和[43]。由于通信时延会影响系统完工时间，因而会考虑任务备份来减少系统完工时间，提高时间效率[44]。关于通信时延方面的软硬件划分更一般研究进展，可以参考文献[45]。

习题

8.1 设计一个方案， 将 9 个任务分到 3 个模块，使得每个模块最少包含 2 个任务，并且使得每个模块的硬件面积不超过 80，计算划分方案后整个系统的性能，使得系统的可靠度最高。模块划分算法使用第 7 章中的 MMM 划分算法，其中链接使用单链接、全链接和均链接各进行一次，并比较结果，给出最好的划分方案。

习题 8.1 通信代价表

	T1	T2	T3	T4	T5	T6	T7	T8	T9
T1		20	12	21	16	15	23	12	17
T2			23	24	26	16	18	20	12
T3				21	21	23	28	22	26
T4					25	23	28	30	22
T5						26	21	13	25
T6							24	25	32
T7								24	25
T8									22
T9									

习题 8.1 多维属性表

任务	软时间	硬时间	硬面积	软可靠度	硬可靠度
T1	30	10	30	0.7	0.9
T2	40	12	44	0.65	0.8
T3	42	10	42	0.7	0.88
T4	35	9	50	0.76	0.91
T5	34	8	45	0.88	0.92
T6	22	6	33	0.89	0.92
T7	23	7	45	0.76	0.88
T8	20	9	51	0.75	0.86
T9	34	12	49	0.90	0.98

8.2 以例 8.3 中图 8-7 任务属性图为例，编程实现基于硬件面积+硬件实现增益的软硬件划分 $HSPA_{AHG}$，并分别在硬件面积约束条件=20 的条件下，进行 2 处理器和硬件（2 并行）软硬件划分，以及 1 处理器和硬件（2 并行）的软硬件划分。（若不能编程实现算法，则进行非编程实现软硬件划分方案，并给出划分过程表）。

8.3 对三个软硬件划分算法 $HSPA_{HG}$、$HSPA_{HGA}$、$HSPAA_{HG}$ 进行实验，从实验结果来判定哪个算法最优？

8.4 在 BUS 总线可以同时进行两个任务数据传输的条件下，对 8.3.6 $HSPA_{GL}$ 算法示例产生的个体 β 进行调度，给出调度过程表和软硬件调度结果以及 β 的适应度和数据传输甘特图。

8.5 阅读文献[45]:Jinyi Xu, Kaixuan Li, Yixiang Chen, Real-time task scheduling for FPGA-based multicore systems with communication delay,Microprocessors and Microsystems 90 (2022) 104468，并撰写阅读报告。