Reading Digest1

马浩铭20337221

A Model-Based Design Methodology for Cyber-Physical Systems, 2011

解决的问题，难点与挑战

在设计信息物理系统(cyber-physical systems)时，我们往往需要使用Model-based design(MBD)的思想。而设计的系统往往十分复杂，涉及到物理系统的数学模型，正式模型的计算，多种不同系统的仿真，软件综合与验证。在这时如果直接使用传统的MBD会变得很困难。

解决的方法与思路

文章提出了一种方法论，将cyber-physical system的MBD设计划分成了十个步骤。这十个步骤顺序并没有固定顺序，但是是相互依赖的。同时这些步骤也是可以不断迭代，直到满足设计要求。

具体来讲，该方法有如下步骤：

1. 申明问题
   这一步中需要使用简单的语言描述设计的目标，并不需要与数学相关，也不需要术语。需要明确对这个系统的特殊要求，比如安全等问题。

2. 对物理过程建模

第一次物理建模迭代需要通过对相关物理系统(例如系统的工作环境或者要控制的物理系统)进行观察，从而建立相应模型，往往需要微分方程或者拉普拉斯变换。

建立的模型也并非一成不变，后续过程中在确定控制算法、确定硬件与测试后，模型也需要不断改进。

3. 确定问题的特征
   对问题中参数进行分类，将固定参数、可调整参数、需要控制的变量分开。确定会影响该物理过程的量、安全限制、输入/输出集。明确物理过程会如何影响系统的计算。

4. 提出控制算法
   确定在什么样的条件下物理过程是可控的，并且提出一个合适的控制算法让嵌入式计算机执行。使用该问题的特征确定各方面的要求。

这一步在后面选择计算模型或者确定硬件后可以再次来到这一步进行迭代。

5. 选择计算模型
   计算模型是指一系列指令，用于计算交互与控制组件的算法。这一步为后续的仿真奠定了基础。通常这一步可以让物理过程能够更加轻松的分析与通过多种方式仿真。许多复杂的cyber-physical system通常需要多个模型用于计算。

6. 确定硬件
   选择能够符合要求的硬件。对于每个部分，要考虑输入与输出的带宽、延迟、能耗、测量精确度与速率、机械参数、耐用性与寿命。选择嵌入式计算机需要对延迟和控制算法执行时间要求的深入理解，并且明确软件如何与一种特定的硬件进行协作。

7. 仿真
   使用仿真工具解决问题。如果选用了多个计算模型，仿真综合工具需要能够让不同模型之间可以互动。

8. 构建
   搭建设备，记录过程中会影响先前建模的部分。构建过程中应该注意要让独立的部分与子系统能够经受理论模型的检验，从而

可以在仿真与测试中迭代。

9. 软件综合
   代码综合有时会被集成到仿真环境中。如果没有现成、好用的代码综合工具，需要确保手写的代码能严格执行选择的计算模型。

就算代码综合器能够严格执行计算模型的语义，并且构建的逻辑也被完全实现，系统的时序行为也需要被确定。其余的限制例如pipelines与caches也需要单独验证。

10. 验证与测试
    调整可调参数来构建尽可能简单测试环境，并且依次单独测试每一个组件与子系统。计算模型需要与物理模型相分离。对执行时间与延迟的测量结果可以被用于改进先前的模型，意外的情况也可能指出模型中的错误。

文章将一个连续的设计过程抽象，分成了十个单独但又相互影响的步骤，明确的每一步中应该做什么，注意点是什么，要求有哪些。这些步骤间没有特定的顺序与次数的限制，不断迭代，直到达到最终设计目标。

为了论证作者的方法模型，作者还提出了一个实例： The Tunneling Ball Device，描述了在这个例子中每一步具体是如何完成的。

其实这样的方法早已被很多工具所使用了。在去年的计算机组成原理的实验中，我们所使用的来自Xlinx的仿真工具就支持这样的方法。能够将复杂的系统分层并抽象，可以单独仿真每一个子系统的时序。完成仿真，确定没有问题后可以进行综合，观察综合的电路图，最后导出到FGPA板上测试。文章将这样的一个大家都在使用的方法明确提出，并且对每一步都做出了明确定义。

An Efficient Heuristic Procedure for Partitioning Graphs

解决的问题

给定一张图，求如何将其划分成几个子图，使得子图大小不超过一个给定的大小限制，且使分割总的代价(切割的边权之和)最小。

问题的难点与挑战

首先定义符号，设图G有n个size为1的节点组成，将要被划分为k个大小为p的子集，所以 kp = n。然后可以由排列组合的公式算出总共有多少种取法。当然这个数量级的复杂度是不可接受的。

这让作者转向了启发式的解法。在设计启发算法时，最重要的是注意算法的复杂度一定要可以接受，任何指数复杂级或者随着节点数量增长的乘积复杂度都不太可能实用。作者直接指定了实用算法的复杂度不超过O(n^2)，n为节点个数。

作者列出了几种现有的解决方法，但都是不实用或者不可用的。

解决方法与思路

首先对于简单情况，需要将2n个节点划分为两个n个节点的子图。设图的价值矩阵为对称矩阵C，边权可以为负。

大致的思路就是开始时将图任意分割为两个大小相等的子图A，B。不断尝试启发改进解，直到无法再次改进为止。

对于具体的改进过程，对于任意非最优解A/B，必定存在X属于A,Y属于B，将X与Y从A与B中交换以后，使得A/B变为最优解。现在的问题在于如何在不遍历所有解空间的情况下确定这个X，Y。

设a属于A，定义E~a~为a的外部总cost，即a到所有A外部点的边权之和。I~a~为a的内部总cost，即a到所有A内部点的边权值和。D~a~ = E~a~ - I~a~

引理1：对于任意a属于A，b属于B。如果将a和b交换，cost会降低 D~a~ + D~b~ - 2 c~ab~

第一版解法：

- 首先计算S中每一个点的D值
- 选择a~i~，b~j~，使得g~1~ = D~a~i~ + D~b~i~ - 2 c~a~i~b~j~ 最大

- 重新计算D'~x~ = D~x~ + 2 c~x a~i~, D'~y~ = D~y~ + 2 c~y a~i~
- 不断重复第二、三步，可以得到局部最优解

最终复杂度为O(n^2^ log n)

后续作者提出了一种解决前部分解法解出局部最优的方案，这是一种基于实践的方法。大致思想就是将通过干扰局部最优解来防止算法停留在局部最优。

具体来讲就是在寻找X与Y时，考虑将A和B分别分解为A~1~/A~2~/B~1~/B~1~,将这四个重新组合，然后再继续使用上述解法。

这样的优化会增加算法的复杂度，但可以接受。并且最差的情况下不会增加总cost。

对于需要分割成两个不等的集合时，考虑引入一系列“dummy”的点，这些点没有任何边与这些点相连。再使用上述算法，这些点会被忽视，最终得到我们想要的结果。

如果由不相等size的点时，可以看作是一群大小为1的点群，再通过很大的边权相连接。

然后对于多分割问题（设划分为k类），核心思想与先前相似，都是先将其划分为k个n个大小的子集，然后进行与上方法相同操作，不过此时遍历一遍的复杂度变为了O(m^2^)，m为点的个数。

为什么能解决问题

其实这篇文章的算法思想和退火、进化算法、梯度下降很相似，都是先生成一个随机解，然后迭代改进该解。因为遍历的复杂度不可接受，而改进的思路是明确的，所以就随便找一个点，然后不断更新优化即可。

该方法也遇到了非凸优化的局部最优解问题，作者提出了一种经验性的解决方案。我认为可以考虑使用退火和遗传算法，考虑迭代过程中并不一定要只有优化时才状态转移，而是可以再没有变优时依概率转移，并且引入初始种群的概念，加入交叉互换。