智能合约和形式验证

在之前一篇关于人工智能的文章里，我指出了“自动编程”的不可能性。今天我想来谈谈一个相关的话题：以太坊式的智能合约的形式验证。有些人声称要实现基于“深度学习”的，自动的智能合约形式验证（formal verification），用于确保合约的正确性。然而今天我要告诉你的是，跟自动编程一样，完全自动的合约验证也是不可能实现的。

随着区块链技术的愈演愈烈，很多人开始在以太坊（Ethereum）的“智能合约语言”上做文章。其中一部分是搞 PL 的人，他们试图对 Solidity 之类语言写的智能合约进行形式验证，号称要用严密的数理逻辑方法，自动的验证智能合约的正确性。其中一种方法是用“深度学习”，经过训练之后，自动生成 Hoare Logic 的“前条件”和“后条件”。

Hoare Logic

我好像已经把你搞糊涂了…… 我们先来科普一下 Hoare Logic。

Hoare Logic 是一种形式验证的方法，用于验证程序的正确性。它的做法是，先给代码标注一些“前条件”和“后条件”（pre-condition 和 post-condition），然后就可以进行逻辑推理，验证代码的某些基本属性，比如转账之后余额是正确的。

举一个很简单的 Hoare Logic 例子：

```
{x=0}   x:=x+1   {x>0}
```

它的意思是，如果开头 x 等于 0，那么 x:=x+1 执行之后，x 应该大于 0。这里的前条件（pre-condition）是 x=0，后条件（post-condition）是 x > 0。如果 x 开头是零，执行 `x:=x+1` 之后，x 就会大于 0，所以这句代码就验证通过了。

Hoare Logic 的系统把所有这些前后条件和代码串接起来，经过逻辑推导验证，就可以作出这样的保证：在前条件满足的情况下，执行代码之后，后条件一定是成立的。如果所有这些条件都满足，系统就认为这是“正确的程序”。注意这里的所谓“正确”，完全是由人来决定的，系统并不知道“正确”是什么意思。

Hoare Logic 对于程序的安全性，确实可以起到一定的效果，它已经被应用到了一些实际的项目。比如微软 Windows 的驱动程序代码里面，有一种“安全标注语言”，叫做 SAL，其实就是 Hoare Logic 的一个实现。然而前条件和后条件是什么，你必须自己给代码加上标注，否则系统就不能工作。

比如上面的例子，系统如何知道我想要“x>0”这个性质呢？只有我自己把它写出来。所以要使用 Hoare Logic，必须在代码上标注很多 pre-condtion 和 post-condition。这些条件要如何写，必须要深入理解程序语言和形式逻辑的原理。这个工作需要经过严格训练的专家来完成，而且需要很多的时间。

自动生成标注是不可能的

所以即使有了 Hoare Logic，程序验证也不是轻松的事情。于是呢就有人乘火打劫，提出一个类似减肥药的想法，声称他们要用“深度学习”，通过对已有标注的代码进行学习，最后让机器自动标注这些前后条件。还在“空想”阶段呢，却已经把“自动标注”作为自己的“优势”写进了白皮书：“我们的方法是自动的，其他的项目都是手动的……”

很可惜的是，“自动标注”其实跟“自动编程”是一样的空想。自动编程的难点在于机器没法知道你想要做什么。同理，自动标注的难点在于机器没法知道你想要代码满足什么样的性质（property）。这些信息只存在于你的心里，如果你不表达出来，任何其它人和机器都没有办法知道。

除非你把它写出来，机器永远无法知道函数的参数应该满足什么样的条件（前条件），它也无法知道函数的返回值应该满足什么样的条件（后条件）。比如上面的那个例子，机器怎么知道你想要程序执行之后 x 大于零呢？除非你告诉它，它是不可能知道的。

你也许会问，深度学习难道帮不上忙吗？想想吧…… 你可以给深度学习系统上千万行已经标注好的代码。你可以把整个 Windows 系统，整个 Linux 系统，FireFox 的代码全都标注好，再加上一些战斗机，宇宙飞船的代码，输入深度学习系统进行“学习”。现在请问系统，我下面要写一个新的函数，你知道我想要做什么吗？你知道我希望它满足什么性质吗？它仍然不知道啊！只有我自己才知道：它是用来给我的猫铲屎的 :p

所以，利用深度学习自动标注 Hoare Logic 的前后条件，跟“自动编程”一样，是在试图实现“读心术”，那显然是不可能的。作为资深的 PL 和形式验证专家，这些人应该知道这是不可能自动实现的。他们提出这样的想法，并且把它作为相对于其他智能合约项目的优势，当然只是为了忽悠外行，为了发币圈钱 ;)

如果真能用深度学习生成前后条件，从而完全自动的验证程序的正确性，那么这种办法应该早就在形式验证领域炸锅了。每一个形式验证专家都希望能够完全自动的证明程序的正确性，然而他们早就知道那是不可能的。

设计语言来告诉机器我们想要什么，什么叫做“正确”，这本身就是 PL 专家和形式验证专家的工作。设计出了语言，我们还得依靠优秀的程序员来写这些代码，告诉机器我们想要做什么。我们得依靠优秀的安全专家，给代码加上前后

条件标注，告诉机器什么叫做“正确安全的代码”…… 这一切都必须是人工完成的，无法靠机器自动完成。

当然，我并没有排除对智能合约手动加上 Hoare Logic 标记这种做法的可行性，它是有一定价值的。我只是想提醒大家，这些标记必须是人工来写的，不可能自动产生。另外，虽然工具可以有一定的辅助作用，但如果写代码的人自己不小心，是无法保证程序完全正确的。

如何保证智能合约的正确呢？这跟保证程序的正确性是一样的问题。只有懂得如何写出干净简单的代码，进行严密的思考，才能写出正确的智能合约。关于如何写出干净，简单，严密可靠的代码，你可以参考我之前的一些文章。

做智能合约验证的工作也许能圈到钱，然而却是非常枯燥而没有成就感的。为此我拒绝了好几个有关区块链的合作项目。虽然我对区块链的其它一些想法（比如去中心化的共识机制）是感兴趣的，我对智能合约的正确性验证一点都不看好。

智能合约是一个误区

实际上，我认为智能合约这整个概念就不靠谱，是一个比较大的误区。比特币和以太坊的系统里面，根本就不应该，而且没必要存在脚本语言。

比特币的解锁脚本执行方式，一开头就有个低级错误，导致 injection 安全漏洞。用户可以写出恶意代码，导致脚本的运行系统出错。比特币最初的解锁方式，是把两段代码（解锁脚本+加锁脚本）以文本方式拼接在一起，然后执行。以文本方式处理程序，是程序语言实现方法的大忌。稍微有点经验的黑客都知道，这里很可能有安全漏洞。

以太坊的 Solidity 语言一开头就有低级错误，导致价值五千万美元的以太币被盗。以太坊的智能合约系统消耗大量的计算资源，还导致了严重的性能问题。

虽然比特币和以太坊的作者大概在密码学和分布式系统领域都是高手，然而我不得不坦言，他们都是 PL 外行。然而如果是内行来做这些语言，难道就会更好吗？我并不这么认为。

首先的问题，是 PL 这个领域充满了各种宗教，和许多的传教士。一般的 PL 内行都会把问题复杂化，他们会试图设计一个自己的“信仰”中完美的语言，而不顾其他人的死活。如果你运气不好，就会遇到那种满嘴“纯函数”，monad，dependent type，linear logic 的极客…… 然后设计出来的语言就没人会用了。

有责任感的 PL 科学家，都是首先试图避免制造新的语言。能不用新语言解决问题，就不要设计新的语言，而且尽量不在系统里采用嵌入式语言。所以，如果换做是我设计了比特币，我根本不会为它设计一种语言。

让用户可以编程是很危险的。极少有用户能够写出正确而可靠的代码，而且语言系统的开发过程中极少可以不出现 bug。语言系统的设计错误会给黑客可乘之机，写出恶意脚本来进行破坏。从来没有任何语言和他们的编译器，运行时系统是一开头就正确的，都需要很多年才能稳定下来。

另外你还要考虑性能问题。对于去中心的分布式系统，这种问题就更加棘手。这对于普通的语言问题不大，你不要用它来控制飞机就可以。然而数字货币系统的语言，几乎不允许出现这方面的问题。

所以与其提心吊胆的设计这些智能合约语言，还不如干脆不要这种功能。

我们真的需要那些脚本的功能吗？比特币虽然有脚本语言，可是常用的脚本其实只有不超过 5 个，直接 hard code 进去就可以了。以太坊的白皮书虽然做了那么多的应用展望，EVM 上出现过什么有价值的应用吗？我并不觉得我们需要这些智能合约。数字货币只要做好一件事，能被安全高效的当成钱用，就已经不错了。

美元，人民币，黄金…… 它们有合约的功能吗？没有。为什么数字货币一定要捆绑这种功能呢？我觉得这违反了模块化设计的原则：一个事物只做一点事，把它做到最好。数字货币就应该像货币一样，能够实现转账交换的简单功能就可以了。合约应该是另外独立的系统，不应该跟货币捆绑在一起。

那合约怎么办呢？交给律师和会计去办，或者使用另外独立的系统。你有没有想过，为什么世界上的法律系统不是程序控制自动执行的呢？为什么我们需要律师和法官，而不只是机器人？为什么有些国家的法庭还需要有陪审团，而不光是按照法律条款判案？这不只是历史遗留问题。你需要理解法律的本质属性才会明白，完全不通过人来进行的机械化执法是不可行的。智能合约就是要把人完全从这个系统里剔除出去，那是会出问题的。

奢望过多的功能其实是一种过度工程（over-engineering）。花费精力去折腾智能合约系统，可能会大大的延缓数字货币真正被世界接受。实话说嘛，试用了多种数字货币，了解了它们所用的技术之后，我发现这些技术相当的有趣，然而这些数字货币仍然处于试验阶段，离真正作为货币使用还有一定距离。集中精力改进它们作为货币的功能，将会加速它们别人接受，而耗费精力去研究智能合约，我觉得是误入歧途。

在这一点上，我觉得比特币比它的后继者们（比如以太坊）都要做的地道一些。比特币虽然也有脚本语言，然而它并不过分强调这种脚本的作用。比特币的脚本语言非常简单，而且不是图灵完备的。这迫使用户只能写出功能简单，不伤害系统性能，容易验证的脚本。相比之下，以太坊花了太多精力去折腾智能合约，弄得过度复杂，搞成了图灵完备的，最终带来各种问题，影响了大家接受最重要的货币功能。