SmartEdge：边缘计算的智能合约

摘要—边缘计算已成为受限设备的有效卸载策略。它使低功能设备可以利用附近的资源来协助执行计算密集型任务。我们设想了一个未来，物联网（IoT）设备可以与其他功能更强大的设备自动进行交易，以请求此类卸载服务。我们相信基于区块链的技术可以通过跟踪使用情况和管理付款来帮助促进这一过程。在这项工作中，我们介绍了SmartEdge，这是一种基于以太坊的边缘计算智能合约，并表明它是用于计算资源的低成本，低开销的工具 管理。

索引词-物联网，边缘计算，智能合约，区块链...

1. **简介**

云计算已经成为一种流行的平台，可以促进来自物联网（IoT）设备的信息的存储和处理[1]。近年来，一种替代方法获得了发展，即边缘计算[2]，这是与本地化数据中心相比，用于在本地或紧邻生成信息的设备上处理信息的一组技术。

最近有关卸载的工作讨论了边缘计算如何管理有限的资源并为移动设备提供有效利用它们的机制[3]。Chen和Xu提出的电信研究[4]探索了小型小区基站（SBS）的用法。作为边缘计算的支持者，可以改善延迟和资源利用的位置意识。然而，由于将来预期的更高需求，需要减轻特定SBS可能需要的高计算量。为此，Chen和Xu [4]讨论了创建这些设备的联盟以共享其资源的机会，其中还包括基于比例的公平效用的基于支付的机制以及用于最小化安全风险的信任网络。相反，我们在SmartEdge上的工作旨在提供一种更简便的方法通过利用智能合约来处理资源分配而无需预先存在关系。

有一些解决区块链和边缘计算的现有工作。Xiong等人的工作[5]涉及使用边缘计算创建移动链，作为医疗，金融等领域先前应用程序的补充。这项研究分析了两种不同情况下的原型模型：一种是固定的采矿奖励与最佳边缘服务价格之间的关系，另一种是可变的，并提供了可在定义最佳资源管理时被这些类型的提供者利用的结果政策。

Stanciu的研究[6]探索了基于区块链的技术的实现，以支持边缘计算领域中的分布式控制系统，使用Hyperledger Fabric生成功能块作为智能合约，这些功能块将由区块链执行并使用以下方法委派任务和资源 Docker容器和Kubernetes用于编排容器执行。

如我们所见，先前的工作已经讨论了边缘计算作为解决物联网系统中存在的计算资源限制的机会。此外，这些研究将区块链作为可行的平台来开发一种可以解决此问题并为系统提供其他功能的机制。然而，据我们所知，这是介绍基于公共无权区块链以太坊的边缘计算智能合约的设计和实现的第一项工作。

1. 以太坊智能合约

术语“智能合约”最早由Nick Szabo提出[7]。它指的是使用软件实现来验证，监视和执行合同，包括转账。虽然可以在原始比特币协议中实现某些简单的智能合约[8]，但以太坊是第一个明确设计为考虑智能合约可编程性的开放式区块链[9]。它被设计为状态转换机，其代码由分散的以太坊虚拟机[10]上的所有矿工同时执行。以太坊原则上允许图灵完备的计算。

我们简要概述了智能合约在以太坊上的工作方式。可以使用称为Solidity [11]的专门编程语言对以太坊上的智能合约进行编码（也有替代方法，但是Solidity是当今最受欢迎的），并可以通过初始交易进行部署。智能合约有其特定的地址。合同由状态变量和功能组成。可以通过寻址该合同的其他交易来调用该功能，以触发状态变量的更改，付款等。为了支持在那里执行智能合同的存储，计算和通信成本 是与每次操作相关的固定“天然气”费用，必须由呼叫者支付。

对于任何交易，都有一个气体限额，该限额应大于该交易所需的气体量（如果该气体限额小于所需量，则发生气体不足异常并且不执行该交易）。此外，每笔交易还必须具有与之关联的汽油价格（以每瓦斯GWei为单位测量），该汽油价格是根据当前市场条件选择的，价格高到足以激励交易快速纳入区块链。粗略地讲，当网络拥挤时，快速包含交易所需的汽油价格会更高。

智能合约具有一定的关键限制，必须牢记。一种是，仅通过发送到合同地址的交易触发它们更改状态，而没有其他任何自动方式来连接链外事件。其次，它们也无法直接与区块链之外的服务器或代码段进行通信，但是它们可以发出事件，这些事件可以使用以太坊API从外部代码进行订阅。最终，由于天然气成本和每个区块的天然气限制（不是固定的限制，而是基于窗口上天然气使用量的移动平均值），因此它们不能任意复杂。

1. Smart-Edge智能合约的设计与实现

我们将智能合约的设计分为三个关键步骤，并在下面的SmartEdge1设计上下文中对其进行描述：

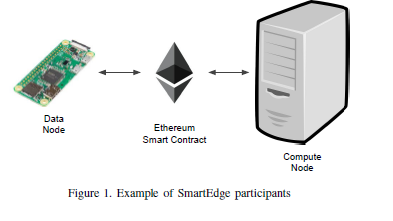
1）确定智能合约涉及的各方

2）确定智能合约生命周期中的关键状态

3）确定并定义触发状态转换的方法

3.1 参与方

要设计智能合约，首先确定交易涉及的各方是有帮助的。SmartEdge合同的设计假设通过以太坊进行交互的参与者有两种类型，如图1所示。第一方是计算节点。计算节点是指将通过SmartEdge合约提供的用于执行工作的计算资源。第二方是数据节点，我们也将其称为交易对手。数据节点拥有需要处理的数据和用于处理它的程序。在本文中，我们将数据和程序统称为工作。



* 1. 状态机

其次，确定智能合约有效期内发生的关键状态非常有帮助。在通过SmartEdge处理的作业的整个生命周期中，合同将经历一系列状态。这些状态如图2所示。每个状态都要求计算节点或数据节点采取措施将合同移至下一个状态。

SmartEdge合同中共有五个状态。我们在这里描述它们：

•不可用：在这种状态下，计算节点不可用于接受任何作业请求。

•可用：计算节点正在宣布其准备分配其资源的价格。它将接受随之而来的第一份工作。

•待处理：计算节点已收到作业请求，正在考虑是否接受它。计算节点必须首先从指定位置检索数据和程序。由URL表示的位置可以引用位于以下位置的文件夹

在数据节点本身或其他一些可访问的资源上

与文件。我们选择使用URL，因为传递给合同的任何内容都将存储在区块链中。将文件直接存储在区块链上是不切实际的，因此我们在此处使用URL作为间接引用。在考虑是否接受作业时，由于可能存在恶意用户，计算节点还可以考虑数据节点的信誉。声誉不是

由SmartEdge合同本身进行监控，但我们

想象一个单独的系统可以用于此

目的。

•正在计算：在这种状态下，计算节点正在忙于处理作业。

•已完成：在此阶段，计算节点已成功完成作业，并正在等待数据节点的确认以结束交互。计算节点将为数据节点提供一个URL，以检索结果。数据节点可以选择接受还是拒绝结果，我们将在后面详细讨论。

3.3 合同方法和事件

第三，重要的是要确定触发状态之间变化的方法（由发送到合同的相应交易调用）。在本节中，我们描述SmartEdge合同的方法。调用这些方法将触发状态转换，如图2所示，并且还将触发可被区块链外部应用程序使用的事件。我们将在第4节中讨论如何在实验中使用这些事件。

首次实例化新的SmartEdge合同时，它将以“不可用”状态开始。通过setAvailable()算节点能够设置用于计算的投标价格。投标价代表每单位时间的固定成本。该价格是在以下条件下设定的：在合同处理的工作期间，将分配已知的固定资源进行处理。由于不一定事先知道要花多长时间才能完成工作，因此我们相信这种定价方案可以对完成的工作进行合理的估价。setAvailable()方法还验证计算节点是否已将足够的资金放入托管账户。计算节点的托管用于评估伪造工作的任何惩罚。托管所需的数量是可以在合同部署之前指定的参数。满足所有可用性条件后，setAvailable()方法会将合同状态更改为Available并发出MadeAvailable事件。

当处于“可用”状态时，交易对手可以开始通过accept-Bid()方法执行工作的请求。调用此方法时，交易对手将发送一个URL，可以在其中找到与该作业有关的文件。这包括数据和处理程序。acceptBid()方法还从交易对手处收取一笔款项，以在进行计算时将其保留在托管中。稍后将从该金额中扣除工作成本。在合同部署之前确定了交易对手托管的最低金额。如果满足托管要求，则合同将进入待处理状态。

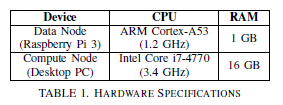
在待处理状态下，合同所有者可以选择接受和拒绝建议的工作。我们包括此功能，使所有者可以评估工作的复杂性和交易对手的声誉。如果所有者不希望接受该工作，则可以调用jobRejected()方法，使合同回到“可用”状态。在这种情况下，将退还交易对手的托管。如果所有者接受该作业，则将调用jobAccepted()方法。此方法将确定作业的开始时间，并将合同发送到“计算”状态。

处于计算状态时，预计计算节点将处理作业。如果由于某种原因计算节点从未完成任务或只是花费太长时间，则交易方可以使用jobTimedOut()方法取消该任务。这种方法将退还交易对手的托管费用，并将合同放回“不可用”状态。在部署之前，已在合同中确定了发生超时之前需要经过的时间。如果作业成功完成，则计算节点将调用jobCompleted()方法。这将确定作业的完成时间，宣布结果的位置，并将合同置于“已完成”状态。

处于“完成”状态时，可能会发生以下三种情况之一。首先，在理想情况下，交易对手会检查结果并感到满意。通过调用resultVerified（）方法，交易对手将支付他们的托管费用，并退还剩余的余额。然后，该合同将恢复为“不可用”状态。其次，如果交易对手认为结果是欺诈的，则可以调用resultRejected（）方法。这将导致交易对手的所有过户押金被退还，合同将进入“不可用”状态。最后，如果交易对方花了太多时间对结果做出响应，则计算节点可以调用resultTimedOut（）方法。计算节点将收到已完成工作的付款，并且交易对手托管的余额将退还。结果超时所需的时间是在部署之前在合同中配置的。与其他两个相关操作一样，合同随后将进入“不可用”状态。

**4.实验装置**

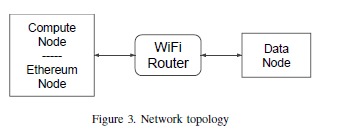
我们的测试平台包括两个组件，如表1所示。我们使用Raspberry Pi 3作为我们的数据节点，并使用功能更强大的台式PC作为我们的计算节点。出于开发目的，我们的计算节点还用作我们的以太坊节点。我们使用以太坊的Truffle开发框架[12]创建了一个测试区块链。如图3所示，两个设备通过WiFi路由器无线连接。



为了使实验期间计算节点和数据节点采取的操作自动化，我们利用合同事件。发出事件后，这些事件将存储在事务日志中，并可用于触发感兴趣的应用程序中的回调。我们使用Web3.js库在测试框架中实现事件回调[13]。清单1中显示了我们的合同中使用的事件的示例。事件是监视智能合约的首选方法，因为事件提供基于推送的通知，而不是基于拉取的通知，在这种情况下，应用程序将不得不不断轮询区块链以检查更改。

**5.结果**

通过一系列实验，我们试图评估提议的智能合约的有效性，并证明它可以支持物联网环境中的边缘计算。特别是，我们旨在评估SmartEdge是否具有以下特性：

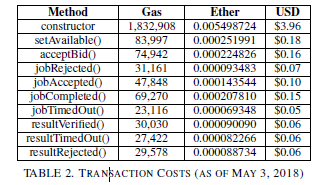


•低开销：由于边缘计算的目的是加快处理速度，因此SmartEdge不应对延迟执行作业做出重大贡献。

•低成本：作为智能合约，使用SmartEdge会产生交易成本。相对于所提供的价值，这些成本不应算太大。

**5.1 交易成本**

表2显示了与使用SmartEdge合同相关的天然气成本。对于计算节点所有者而言，SmartEdge合约将花费约3.96美元部署在以太坊区块链上。我们认为这是合理的合同前期费用。该合同是可重复使用的，因此所有者最终应该能够收回部署成本。其余的合同方式都便宜一个数量级，每次通话的费用为$ 0.18美元或更少。



**5.2 高架**

为了评估SmartEdge的开销，我们首先创建了要测试的工作负载。我们开发了一项工作，涉及逐行从文件中读取整数并确定它们的乘法因子。输入文件包含从1,000到2,000,000,000范围内随机选择的数字。此文件的小版本（input.txt）由10,000个整数组成，而大版本则由100,000个整数组成。python脚本（factor.py）使用pyprimes库确定因素。表3中显示了所有文件的文件大小。还显示了与每个输入文件相对应的结果文件。

我们使用数据节点，计算节点和SmartEdge测量了分解作业的执行时间。结果显示在表4中。在Raspberry Pi 3上执行该作业需要3分钟以上的时间。但是，当使用SmartEdge执行该作业时，只需要8.6秒。与直接在计算节点上执行作业相比，有2秒的开销。应当注意，这2秒的开销包括将作业转移到计算节点和将结果转移回数据节点所花费的时间。

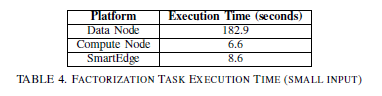


图4显示了SmartEdge合同在每个状态花费的时间百分比。不出所料，计算状态主导着时间，这是期望的行为。但是，我们意识到在Truffle开发区块链中，区块是即时开采的，实际上这并不是采矿时间的准确表示。我们使用Truffle的Ganache区块链重新进行了实验，挖掘时间设置为以太坊平均14秒。结果如图5所示。在此图中，我们可以推断出每笔交易的挖掘时间占合同每种状态所花费的时间的主导，并且是选择卸载工作时的重要考虑因素。

我们又进行了一次实验，以查看如果作业的执行时间相对于块挖掘时间更长，则持续时间将受到怎样的影响。直接在计算节点上运行时，输入文件较大的作业的执行时间为67秒。实际上，我们可以在图6中看到，相对于其他状态，在计算上花费的时间百分比显着增加。

我们认为，较大的输入文件和结果文件将分别对“可用”和“已完成”状态产生相似的影响，因为它们会影响文件传输时间。我们的目的是在以后的工作中研究这个假设。

**6.风险与安全性问题**

我们已经在下面确定并讨论的当前合同设计中存在一些问题。

结果验证：由于智能合约的计算限制，我们认为通常很难通过智能合约本身来验证作业是否正确执行。尽管有可验证的计算技术[14]，但电路的昂贵成本，在这种情况下，生成和同态加密通常使其不切实际。现在，我们将其留给数据节点以验证结果是否有效。如果检查所有结果都很简单，那么数据节点可以这样做。但是，如果结果验证也是计算密集型的，那么数据节点可以将一些测试数据随机地植入原始输入文件中，对于该结果而言，结果是已知的。例如，如果我们假设与实验中的工作结构类似，则数据节点可以将测试数据植入到输入文件的第n行。或者，只要数据节点可以跟踪放置情况，就可以以更加随机的方式放置测试数据。然后，数据节点可以将这些特定值的结果用作代理来确定其余工作是否确实正确执行。

另一方面，数据节点可能是恶意的，因此计算节点可能处于危险之中。恶意数据节点可能会拒绝结果，以避免必须为执行其任务而付费。对于此问题，我们认为基于信誉的系统可能会有用。计算节点可以在接受数据作业之前检查数据节点的信誉，以避免这种情况。另一种可能性是作业本身是出于恶意目的而设计的。但是，我们设想通过SmartEdge处理的所有作业都将在沙箱中执行，例如虚拟机或容器，以便将其包含在主机系统中并与之隔离。

**数据交换：**SmartEdge当前设计中的另一个安全问题是数据节点与计算节点之间信息的安全传输。当前，URL用于指定文件的位置。一旦此信息进入智能合约，它将成为可公开访问的信息。如果正在交换的数据很敏感，则可能会出现问题。

**7.未来的工作**

我们希望将一些可能的改进合并到我们的SmartEdge实施中。

**数据隐私**：如第6节所述，可能需要在数据节点和计算节点之间安全地传输信息。解决此问题的简单解决方案包括以某种方式对URL进行加密，以使密钥仅在每个工作的参与者中才是已知的。

**出价曲线**：在当前版本的SmartEdge中，计算节点的出价是固定的。但是，我们认为有用的修改是将出价定义为完成时间的函数。这种变化的好处是，一方面，它将激励计算节点尽早完成工作；另一方面，它将减少递归收益或对后期完成工作的惩罚。

**拍卖合同**：当前系统的另一个可能的改进是创建单独的拍卖合同，该合同可以自动将数据节点与最合适的计算节点进行匹配。当前，数据节点直接与计算节点打交道，但是如果它们想货比三家并考虑多种资源，拍卖合同可以自动为数据节点管理该过程。

**8.结论**

我们介绍了SmartEdge，这是一种新的基于以太坊的边缘计算智能合约。它允许节点以可验证的方式将计算分担给其他方的边缘计算设备，以换取付款。我们相信SmartEdge将成为物联网应用的宝贵工具。尽管我们当前的实现只是概念验证，但我们的目标是继续在SmartEdge上进行开发。