**基于区块链的汽车分时租赁交易隐私保护方法**

**技术领域**

本发明涉及到区块链系统交易，以及交易各方的隐私保护领域，形成了一种基于区块链的汽车分时租赁交易隐私保护方法。

**背景技术**

一、共享汽车的发展

随着交通运输部、住房城乡建设部制定的《关于促进小微型客车租赁健康发展的指导意见》出台，明确鼓励共享汽车发展，为行业扫清了政策障碍。近日，来自普华永道的统计数据也显示，未来5年汽车分时租赁（共享汽车）市场将以超过50%的增幅继续发展，保守估计在2020年全国车队规模将达到15万台。

二、区块链在共享汽车中的应用

近年来随着汽车共享概念逐渐落实与普及化，电子移动技术也正在推动交通领域的发展，将区块链技术引入汽车移动空间，从充电站到支付解决方案，都接入了区块链，更好的保护了双方的账户隐私信息，实现减少拥堵和污染、缩短通勤时间和降低运营成本。

区块链技术在汽车共享上的应用，旨在利用区块链技术降低汽车共享平台的使用成本，可用来记录车辆的所有权以及状态参数，并且处理某些特定类型的金融交易。

区块链是一种分布式、去中心化的运营系统，具有安全、可靠、不可篡改等特点，在数字货币交易领域具有广泛的普及应用。在区块链系统中通过应用零知识证明（zkp）等密码学技术，区块链可以很好地隐藏掉各方用户的交易记录和身份信息等相对敏感数据，从而保护了用户的个人隐私。此外，区块链作为一种底层技术，还可以支持智能合约。智能合约是一种安全可靠、自动化运行的程序，不手第三方约束，一旦触发智能合约，就按照里面的逻辑顺序一次执行，保证了公平、公正、公开，理论上可以为用户提供任何计算服务。包含智能合约的区块链系统在金融、医疗、能源、保险和物联网领域都有着广泛的应用。

在汽车分时租赁交易平台中，利用原始的第三方租赁交易平台存在着很多难以解决的弊端：交易信息不公开透明；所有信息都掌握在第三方平台手中；容易篡改数据；产生彼此信任问题。而区块链技术可以很好地解决这些弊端，它是一个去中心化的平台，所有交易数据信息公开透明，不可篡改。并且可以解决交易双方的彼此信任问题。

**发明内容**

1. 如果你附近的人拥有空闲的车辆，可以进行汽车租赁服务交易，那么你就可以在区块链对等网络中轻松获得该服务，与其进行租车交易，这将节省时间和其他费用,包括巨额交易费用。
2. 用户信息将使用加密技术来保护；车辆状态参数信息，车辆使用情况所有公开数据加载到区块链平台中，隐私信息进行相应隐藏，旨在保护租车用户信息等隐私内容。
3. 车辆状态参数信息可通过全球定位系统（GPS）获得，并将这些参数返回给Car保存；汽车上有二维码，用户想要租车时扫描二维码，汽车使用状态信息都将展现给租车用户。
4. 本基于区块链平台的租车方案可以解决信任问题，一旦将交易记录在一个块中，其他人就无法对其进行篡改。方案还可以保护交易双方的账户隐私信息，使交易中没有任何关联性，敏感信息不被泄露，与传统的交易平台不同，使用户在租车过程中可以完全隐藏自己的个人账户信息和交易金额，但又可以正常完成租车结账等服务。
5. 本发明中租车用户通过初始化一笔交易押金，即可享受租车服务。不同租车用户通过分时共享，按照自己的实际租车距离来分配租车费用。租车用户享受完租车服务，通过区块链交易以及事先部署好的智能合约完成结算。整个交易结算过程在区块链系统中被拆分为不同类型的交易，这些交易通过零知识证明技术zk-snark为每笔交易提供有效的证明，使交易每笔交易在区块链系统中都被正常验证和接收打包成区块上链。这些操作无法获知交易的发送方，取消了交易和交易发送者的关联，从而保护了用户的隐私信息。本设计方案的优点是即保护user账户信息隐私和交易金额，又可以公平划分不同user的租车费用。

**术语解释**

zk-snark技术：一种零知识证明系统，首字母缩略词zk-SNARK代表“零知识简洁非交互式知识论证”，并且指的是一种证据构造，其中人们可以证明拥有某些信息，例如秘密密钥，而不会泄露该信息，并且之间没有任何交互。证明者和验证者。“零知识”证明允许一方（证明者）向另一方（验证者）证明陈述是真实的，而不会泄露超出陈述本身有效性的任何信息。例如，给定随机数的散列，证明者可以说服验证者确实存在具有该散列值的数字，而不会揭示它是什么。

零知识余额/金额：一种使用哈希值表示的隐藏余额/金额，其组成形式为：cmt=SHA256(v, sn, r)，其中v是cmt对应的明文金额数值；sn是cmt唯一对应的序列号，sn公布代表cmt已被花费；r是用于防止暴力破解的私密随机数。零知识余额的使用和存入必须通过零知识证明验证，使用新的零知识余额代替旧的零知识余额。一旦零知识余额更新，旧余额的序列号sn会被公布。

**技术方案**

方案设计：

（1）本方案大致分为两种汽车分时租赁的情况：①在汽车租赁起始点所有user上车；②经过沿途线路时有user上车（汽车有空座位的情况下）

---》》》本方案中汽车Car上装配的智能设备记录每个user用户租车服务的距离dist\_i（每个user的租车距离），将数据dist\_i，Σdist\_i（所有user租车距离之和）和汽车行驶总距离dist传给user和owner用于计算租车费用cost以及结算和划分租车费用；

---》》》汽车拥有者Owner根据Car实际被租用距离dist利用计算规则函数Calculation Rules()计算出总租车费用cost——所有user总共需要支付的租车费用，线下所有user根据自己实际租用距离划分租车费用cost；

---》》》汽车使用者User根据dist\_i,Σdist\_i,cost等已知参数计算出自己在本次服务中应付的租车费用cost\_i=cost×dist\_i/(Σdist\_i)；

---》》》智能合约接收Owner发送的cost，dist\_i和Σdist\_i等参数并计算user实际需要支付的租车费用并验证Car user自己计算的cost\_i的正确性。

（2）详细方案：

①owner<<<<<<<<<<<初始化：

Car owner提前规定共享汽车的最大行驶里程N，租车垫付费用押金fees（Car user需交付的租车押金，其对应最大行驶里程N），owner部署智能合约。

Car user的数量可根据共享汽车可乘载乘客数量而定，Car user在自己当前的可租车地点判断共享汽车目前可租用里程是否满足自己的总租车里程后再进行租车，每个user上车前需要在自己的账户中扣除租车押金（fees）用于租车结算【Cost】，所有user共享同一辆汽车的租车服务，最后一位user的租车服务结束后，每个user根据自己的实际租车里程（dist\_i）占所有user总租车里程和(Σdist\_i)的比例进行费用结算【Divide】。

整个过程可通过Car的智能终端设备记录共享汽车的行驶位置信息以及参数，以及每个user的行驶里程dist\_i——用于单个user计算租车费用，Car的智能终端设备记录每个user的dist\_i以及Σdist\_i并传给user\_i和owner， user\_i和owner通过与智能合约交互验证cost\_i和cost的正确性，完成各自费用的正确结算。

②user/owner<<<<<<<<<<<租车费用划分：

cost（cost<=fees）是owner通过最终所有user租车服务结束完，Car实际行驶距离dist计算得来，是所有user总共需要支付的租车费用。

cost=Calculation Rules（time(start/end) | location(start/end) | dist）

#owner：领取租车费用cost到自己的账户

#user：根据单个user租车里程所占所有user总租车里程划分总租车费用cost，单个user\_i的租车费用计算方法：**cost\_i=cost×dist\_i/(Σdist\_i)**，其中Σdist\_i为所有user各自乘坐的里程相加之和。

③user<<<<<<<<<<<剩余租车费用回退到user账户

user与智能合约交互，智能合约判断refund\_i的正确性，

refund\_i=fees - cost\_i

如果智能合约验证refund\_i有效，则user可将其加入到自己账户余额中。

**#####以下为方案各步骤的详细操作#####：**

***Deploy-sc***：

Car owner提前规定共享汽车的最大租车行驶里程N，租车需抵押费用fees--对应最大行驶里程N，然后owner部署智能合约。

***Cost：user***

Car user进行租车服务前，先从自己的零知识账户中扣除fees数量大小的押金费用用于租车服务，并更新自己的零知识余额即：Car user从原来自己的零知识账户余额cmtU中转化一笔零知识金额cmts，其对应的明文金额为fees,Car user生成一个零知识证明πcost，证明如下内容：

①原来的零知识余额cmtU= SHA256(vU | snU | rU)；

②要支付的零知识金额cmts= SHA256(fees | sns | rs)；

③操作后的零知识余额cmtU’= SHA256(vU’ | snU’ | rU’)；

④vU大于等于fees，vU-fees= vU’；

其中公开的参数为snU，cmtU，cmts，cmtU’；

私密的参数为vU，rU，fees，rs，vU’， snU’， rU’，sns；

Car user将公开参数与πcost，发送到区块链中，矿工使用公开参数验证πcost的正确性，将交易记录在区块中。Car user的零知识余额更新为cmtU’。

***Commit，claim，divide操作在ethereum需要与智能合约进行交互！！！***

***Commit：user***

Car user调用智能合约Init函数并初始化所需参数（fees,subcosts,subdists），证明fees对应的零知识金额为cmts并给智能合约初始化账户余额。~~（用零知识余额充当押金必须要证明）~~

生成一下证明：

①cmts= SHA256(fees | sns | rs)；

②零知识金额cmts存在于以RTcmt为根的默克尔树中。

***Declare：owner（声明自己要取走的钱cost）***

Car owner通过Car实际被租用距离dist利用计算公式求出总租车费用cost，以及用于绑定cost的commit hash-cmtt，为了保证owner和user在用同一个cost进行结算，并线下和所有user共享cost和cmtt。

owner上传cmtt凭证，并生成下面的证明：

①cmtt=SHA256（cost | r）

②cmts=SHA256（cost | sns | rs）;

③cost = 5 × dist

并向智能合约发送cost，cmts参数。智能合约验证Car owner前后所提供的cost是否相等，如果相等，则证明cmts有效，用于后面owner Collect。

***Divide：user（声明自己要取走的钱refund\_i）***

所有Car user租车服务结束后，Car上的智能设备将记录每个user的dist\_i，将数据dist\_i和Σdist\_i传给user和owner，将dist传给owner；

Car user利用得到的参数计算生成一个零知识证明πdivide，证明如下内容：

①cost\_i=cost×dist\_i/(Σdist\_i)

refund\_i = fees - cost\_i

②cmtc=SHA256（refund\_i | sns | rs）

③cmtt=SHA256（cost | r）

其中公开的参数为cmtt,cmtc,dist\_i,Σdist\_i,fees；

私密的参数为cost，cost\_i，refund\_i,sns，rs，r；

Car user将公开参数与πdivide，发送到区块链中，并向智能合约发送refund\_i进行验证，智能合约根据user，owner提供的参数会自行计算user的cost\_i以及refund\_i。智能合约验证通过后，cmtc成为一笔有效的零知识金额,用于后面user Refund；矿工使用公开参数验证πdivide的正确性，将交易记录在区块中。

***Collect*：owner（*把钱取走*）**

Car owner 收取租车费用cost=5 \* dist到自己的账户，并更新零知识余额，生成一个零知识证明πcollect，证明如下内容：

①cmtt=SHA256（cost | r）

②当前的零知识余额cmtO= SHA256(vO, snO, rO)；

③要存入的零知识金额cmts=SHA256（cost | sns | rs）;

④操作完成后零知识余额cmtO’= SHA256(vO’, snO’, rO’)；

⑤vO+cost = vO’

其中公开的参数为snO,sns,cmtO,cmtO’,cmtt;

私密的参数为vO,rO,cost,r,rO,rs,vO’,snO’,rO’,cmts; 注：cmts用默克尔树进行混淆保护

Car owner将公开参数与πcollect，发送到区块链中，cmts由于sns的公开而失效，Car owner的cmtO’更新。

***Refund：user（把钱取走）***

Car user领回剩余押金到自己的账户,refund\_i = fees - cost\_i，user将refund\_i和cmtc发给智能合约进行验证，确保refund金额的准确性。并更新自己的零知识余额为cmtU’

①refund\_i = fees- cost\_i

②原来的零知识余额cmtU= SHA256(vU | snU | rU)；

③要转化的零知识金额cmtc= SHA256(refund\_i | sns | rs)；

④操作后的零知识余额cmtU’= SHA256(vU’ | snU’ | rU’)；

⑤vU + refund\_i = vU’；

其中公开的参数为：snU，sns，cmtU，cmtU‘，fees；

秘密的参数为：vU，rU，rs，vU’，snU‘，rU’，cost\_i，refund\_i，cmtc;cmtc用默克尔树进行混淆保护

Car owner将公开参数与πcollect，发送到区块链中，Car user的cmtO’更新。

**本发明的有益效果为：**

1通过租用共享汽车user可以节约大量的租车费用，共同租车人数越多，实际单个user支付费用相对越少；

2可以隐藏user的租车里程和租车费用，保护各方账户和消费隐私；

3不同user在区块链平台通过智能合约进行租车费用结算，租车服务结束费用划分自动执行，防止user抵赖，保护了owner的切实利益，以达到公平公正。

**附图说明**

**......**

**具体实施方式**

**实例:**

在本实例中有user用户和owner用户，owner用户拥有汽车，提供租车服务，user用户享受租车服务。

①owner用户提前规定共享汽车的最大租车行驶里程N，租车需抵押费用fees--金额对应最大行驶里程N，与user用户共享。

②不同user用户若满足租车条件，扣除押金享受租车服务，并更新账户的零知识余额cmtU’= SHA256(vU’ | snU’ | rU’)，并将公开参数与πcost，发送到区块链中，矿工使用公开参数验证πcost的正确性，将交易记录在区块中。

③所有user用户租车服务结束后，计算自己所支付的费用cost\_i，并计算cmtc=SHA256（cost\_i | sns | rs），将公开参数与πdivide，发送到区块链中，并向智能合约发送cost\_i进行验证。智能合约验证通过后，cmtc成为一笔有效的零知识金额；并从user以智能合约的形式初始化的fees押金中扣除租车费用cost\_i。

④owner 收取租车费用cost到自己的账户，并更新零知识余额cmtO’= SHA256(vO’, snO’, rO’）将公开参数与πcollect，发送到区块链中，并向智能合约发送参数用于验证。

⑤user领回剩余押金到自己的账户，并更新自己的零知识余额cmtU’= SHA256(vU’ | snU’ | rU’)

Smart Contract：

**Init**：

①owner在区块链中部署智能合约

②user向智能合约存入coin明文押金，然后才能开始租车服务

function init（coin）

Set:coin

Set:Collected = false

Set:......

If start of service then

Lock coin from user

End if

End function

**Cost**：

③user服务完成将计算的para传给智能合约用于collect验证（cost）

function cost（para）:

Set:Para = para

End function

**Collect：**

④owner自己计算应得费用的零知识金额cmtf以及para验证传给智能合约（collect）

function collect（para，cmtf）：

If end of service | Para equal to para | Collected = false

Set:cmtf valid

Set:Collected = true

Release coin to user

End if

End function

⑤一次服务结算后智能合约立即销毁