1. 尊敬的主席，尊敬的专家，亲爱的同行们：大家早上/下午/晚上好。我非常荣幸能够在这个重要的学术会议上，向大家展示我的研究成果。我是施宸昊，来自上海交通大学，。本次演讲的目标是探讨R1CS范式生成的相关问题，并提出一个新的算法来应对范式生成的问题。
2. 首先，我将介绍相关的研究背景。然后，我将通过展示一对等价R1CS在算法过程中的中间输出，详细阐述所提出算法的具体步骤。然后，我将介绍实验中所使用的数据集以及实验结果。最后，我将总结我的研究的独特贡献，并且讨论目前我的研究的局限性以及future work。
3. 首先，让我介绍一下论文的研究背景
4. （介绍零知识证明）零知识证明是一种可以证明某个事实或声明的方法，而不必透露与该事实或声明相关的任何其他信息。它的基本原理是让证明者通过一系列计算，向验证者证明自己拥有某种知识或权限，而不需要透露具体的信息。（Primary Application）在这个数字化时代，个人信息已经成为了一种可被商业化和利用的商品，而随着技术的进步，这种信息泄露的风险也随之增加。越来越多的密码学社区试图解决一些区块链最重要的挑战：隐私和可扩展性。（Key Application Domains）除此之外，区块链也被广泛应用于Web3， 加密货币等重要领域。（complex transformation process）。但是，零知识证明并不能直接被应用于现实中的任何计算问题。我们需要将现实中的计算问题首先转化为算术电路，然后将算术电路转化成R1CS，再到QAP，最后再为这个QAP创建实际的零知识证明。
5. 在零知识证明的底层工具链中由电路语言到R1CS约束这一步转换存在着很多局限，R1CS描述了使用高级编程语言编写的语句的执行方式，并被许多零知识证明应用程序所使用，但目前没有标准的表示方法。（Limited Mergeability）这就导致R1CS的可合并性较差。由代码段A与代码段B合并后所生成的R1CS与代码段A和代码段B独立生成的两个R1CS从形式上看可能毫无关联，这也为我们验证所生成R1CS的正确性带来了很多困难。（Flexible R1CS Representations）这与R1CS本身表达能力局限性有关外，根本原因便是相同语义的程序本身可以生成多个等价的R1CS约束，R1CS约束组中约束之间的合并与拆分也会导致R1CS在形式上发生变化。（研究意义）所以我们需要在等价的R1CS约束组中提出R1CS约束组的范式构造方式，使得对于不同的R1CS约束，我们可以较容易地判断其等价性和正确性。这对我们验证程序的等价性以及正确性，包括后续更加深入研究R1CS的可合并性方面都将大有裨益。
6. 下一个部分是算法流程设计。这个部分讲述了本论文所提出算法的具体流程。
7. 在这个部分中，我将通过一对等价R1CS约束组在算法中各个阶段的中间输出来介绍算法的具体流程for better understanding。 R1CS was introduced in V神的博客，对应相同的Circom电路
8. （Main Step）将每一个约束R1CS约束需要满足的等式转化为算式，再将这些转化出的算式结合在一起，就得到了一个以DAG形式存储的含有公共子式的算式树。在算法中，我们依次转换约束，为其中新出现的变量新建结点，重复利用在前面约束中已出现过的结点。
9. （left 上方黑字）在算法中，我们使用一种特殊的数据结构，RNode，来存储R1CS变量的信息。 任何一个RNode既能被视作一个简单的变量，也能被视作以其为根节点的算式子树的结果。这样的设计符合R1CS中约束拆分与合并时的特点，从而也将R1CS中约束的分解和合并带来的不同尽可能降低。（left 灰字）在R1CS中，约束的合并和拆分会为R1CS带去诸多影响，比如，中间变量的数量和选择， 约束的数量以及形式，以及variable mapping in particular R1CS compiler。但是根据这两个算式的比较，我们可以发现这些变化在算式中并没有非常明显的体现。（right）在这一步中，等价R1CS所生成的算式树之间仍然存在不同，这种不同来自于R1CS约束组中的连加式。在这一个步骤中，我们的算法没有足够的信息去确定连加式中变量的相加顺序，而只是简单的遍历输入的R1CS。因此，需要进一步的abstraction
10. 第二个步骤是瓦片选取过程。在这个步骤中，我们将上一个步骤生成的数据流图分割为自定义类型的瓦片。我们设计了三种瓦片类型：Quadratic， MulLinear and AddLinear。我们仅仅设计了最简单的几种瓦片 for three consideration。首先，我们首先暂时搁置了约束间合并的步骤，待后续步骤获取树中更多信息后再进行，其次，瓦片选取步骤的算法实现较为简单，最后，将合并的步骤建立于未合并的范式的基础上，简化该步骤算法的设计。
11. 瓦片选取的过程如下图所示。
12. 在第三个步骤中，我们将Linear 瓦片进行抽象，来为等价的R1CS生成相同的抽象后的数据流图。As mentioned in the graph-constructing step，等价的R1CS约束组所生成的数据流图的不同之处在于，在处理线性瓦片时节点之间的相加顺序不同。但如果将所选取出的线性瓦片中的相加的节点视作一个集合，他们又是相同的。换句话说，相加顺序的不同，在瓦片选取的过程中仅仅意味着将各节点加入线性瓦片的顺序的不同。

在这个步骤中，我们将线性瓦片抽象成一个大的抽象节点，对不含于该瓦片的外部节点，屏蔽了具体的内部结构。这样一来我们就屏蔽了线性瓦片内部具体相加顺序的不同的差异，让外部节点到线性瓦片中具体节点的联系, 变成到这个节点具体所属的瓦片的联系。从而得到一致的数据流图。

抽象之后的数据流图与抽象前相比，结构更加简单。边的类型有以下几种：

* 非线性瓦片抽象节点到非线性瓦片抽象节点: 两个顶点在抽象前的数据流图中便已经存在。与抽象前的数据流图保持一致。
* 非线性瓦片抽象节点到线性瓦片抽象节点: 当且仅当抽象节点所代表的线性瓦片中存在非抽象节点时存在。
* 线性瓦片抽象节点到线性瓦片抽象节点: 当且仅当两个抽象节点所代表的线性瓦片存在公有的非抽象节点时存在。

1. 在本步骤中，使用加权PageRank算法计算出抽象后的数据流图中各结点的权重，进而以结点的权重为基础计算瓦片相对应的约束的权重。相对于传统的PageRank算法，该算法为图中的每一条边都赋予了权重，并调整了节点权重的迭代公式。

本算法使用加权PageRank的主要目的，是为了降低抽象后的数据流图的对称性。由于前一步骤中，对线性瓦片进行了简化，导致数据流图的结构得到了大幅度的简化，在图中的部分结构上存在对称的情况。如果用一般的算法计算图中节点的权重，在图结构中对称存在的节点，容易计算出相同的权重，对后续对约束以及排序的问题造成困扰，所以利用加权PageRank算法为不同的节点计算出权重，增加图的不对称性，从而尽可能避免相同得分的出现。

1. （Adjustment of Scope）本步骤中对只在线性约束中出现过的变量进行排序。

在瓦片划分步骤中，确定了R1CS范式中约束的划分；在瓦片权重计算的步骤中，确定了R1CS范式中约束的排序方式；在前一步骤中，确定了在二次约束中出现过的变量在变量映射中的位置。因此，距离最终的R1CS范式生成，只剩下那些只在线性约束中出现过的变量在变量映射中的位置还没有确定。这也是正常的现象，因为在对数据流图的抽象步骤中，消去了线性变量的内部具体结构对整体数据流图结构的影响。所以在之前的步骤中，没有足够的信息对其中的变量进行排序。

（Sorting Criterion）对于这类变量，本文提出了新的权重排序方式，公式如图所示

也就是说在每一个新加入的线性瓦片中，对于每一个新引入的变量，其权重是除去本身线性瓦片以外的其他线性瓦片中该变量的系数和该瓦片权重的乘积的绝对值之和。

1. 接下来的部分是evaluation，这个部分要介绍了本论文中所设计的测试集，并且展示了所实现算法在数据上的测试效果。
2. （benchmark ck）由于相关研究的缺失，目前该领域并没有一个非常完备的测试集。于是我们通过在算法测试过程中发现的一些规律，对等价R1CS约束组的生成进行了更加细致的归纳与划分，并根据得出的规律设计出了一个较为完备的测试组。

（Separated Generation）测试组中的每个类别包含2到3个基础的R1CS约束组。为了更全面地测试算法的鲁棒性和正确性，根据类别所对应的原因对每个约束组都生成了5到6个与之等价的R1CS约束组。每个约束组的等价约束组都被成对地输入到算法中进行测试，以验证算法在处理不同的等价约束组时是否能够生成一致且符合前述R1CS范式定义的输出结果。

1. 最后一个部分是conclusion。主要介绍了本文的主要贡献以及future work 。
2. This paper proposes a data-flow-based algorithm for constructing the normal form of a group of semantically equivalent R1CS, a high-level programming language used for ZKP. And design a dataset for equivalent R1CS
3. Future work includes establishing rules for merging constraints and conducting a more comprehensive benchmark. This requires us to conduct more in-depth research and exploration of the generation rules of R1CS to improve the algorithm.
4. 感谢各位的聆听与参与。通过这次汇报，我们希望能够增加对R1CS的理解，为未来对R1CS正确性验证以及合并性提供有价值的信息。感谢各位的关注和支持。如果您有任何问题或意见，我很愿意与您进一步讨论。谢谢大家！

Dear Chairman, esteemed experts, dear colleagues: Good morning/afternoon/evening. I am honored to present my research findings at this important academic conference. My name is Chenhao Shi, from Shanghai Jiao Tong University. The goal of my presentation is to explore the relevant issues in generating R1CS paradigms and propose a new algorithm to address the challenges in paradigm generation.

Firstly, I will introduce the background of the research. Then, I will illustrate the specific steps of the proposed algorithm by showcasing the intermediate outputs of a pair of equivalent R1CS during the algorithm process. Next, I will introduce the dataset used in the experiments and present the experimental results. Finally, I will summarize the unique contributions of my research and discuss the current limitations and future work.

First, let me introduce the research background of the paper. Zero-knowledge proofs are a method for proving a fact or statement without revealing any other information related to that fact or statement. The fundamental principle is to allow the prover to prove possession of certain knowledge or privileges to the verifier through a series of computations without disclosing specific information.

In this digital age, personal information has become a commoditized and exploitable asset, and with technological advancements, the risks of information leakage have increased. The cryptographic community is increasingly attempting to address some of the most critical challenges in blockchain: privacy and scalability. Additionally, blockchain is widely applied in important domains such as Web3 and cryptocurrencies.

However, zero-knowledge proofs cannot be directly applied to any computational problem in reality. We need to first transform real-world computational problems into arithmetic circuits, then convert the arithmetic circuits into R1CS, further into QAP, and finally create the actual zero-knowledge proofs for this QAP.

There are many limitations in the conversion from the circuit language to R1CS constraints, which is a critical step in the underlying toolchain of zero-knowledge proofs. R1CS describes the execution of statements written in high-level programming languages and is used by many zero-knowledge proof applications. However, there is currently no standard representation for R1CS. This leads to poor mergeability of R1CS. The R1CS generated after merging code segment A with code segment B may have no apparent relationship with the two independently generated R1CS from code segment A and code segment B. This poses significant challenges for verifying the correctness of the generated R1CS.

The limited mergeability is related to the expressive limitations of R1CS itself. The fundamental reason is that programs with the same semantics can generate multiple equivalent R1CS constraints. The merging and splitting of constraints in an R1CS constraint group can also cause formal changes in R1CS.

Therefore, we need to propose a paradigm construction method for R1CS constraint groups that are equivalent. This will facilitate the determination of equivalence and correctness for different R1CS constraints. It will be beneficial for verifying the equivalence and correctness of our programs and further research on the mergeability of R1CS.