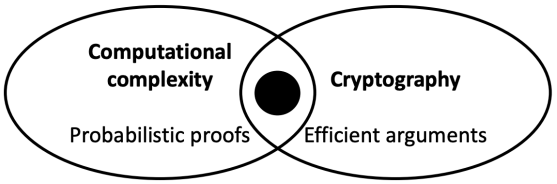
06-Building_SNARKs-note

SNARKs

- Cryptographic proofs for computation integrity that are super **short** and are super **fast** to verify.
- Origins in the 1990s:



2

PPT第2页

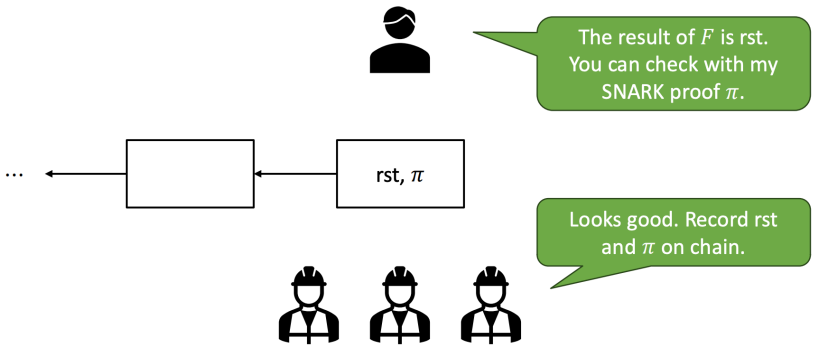
对于完整性和健全性两性质，老师举例：

如果P试图向V证明F∈L，这里L就是某个NP语言（比如3SAT或哈希预映像等），

completeness：如果F∈L，那么V一定输出accept。

soundness：如果F∉L，那么V一定输出reject。

Blockchain Application



3

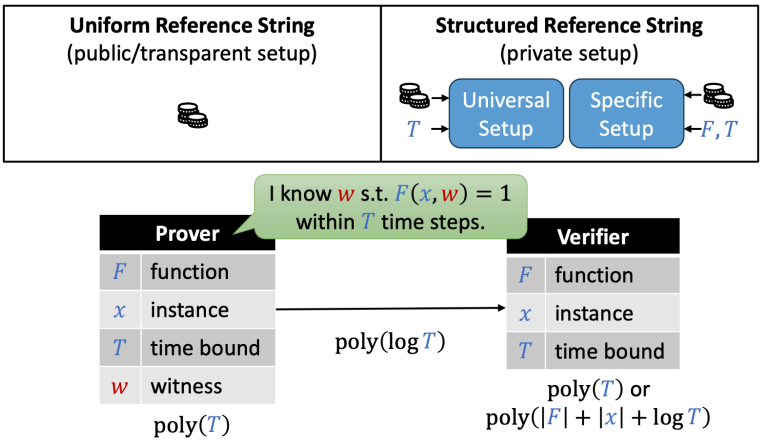
PPT第3页

ZK在链上的意义：

某个prover计算了s.t. f(x,w)=1，将这个proof在链上发布。随后这个proof可以在链上被多方验证。整个计算过程只由一个prover执行了一次，且验证proof比计算f更快，省gas且cheap，并且保证了计算的完整性。

SNARGs

- Succinct Non-interactive ARGuments (SNARK: SNARG of Knowledge).



4

PPT第4页

SNARKs的各字母依次含义：
Succinct：small proof size和small verification time。

Non-interactive：证明者只需一次性发送证明，不需要和验证者进行多轮交互（非交互式的意义：不需要verifier一直在线）。

Argument：基于计算假设（如离散对数/LWE）而安全，允许有可忽略的小失败概率（不是严格的proof）。

Knowledge：证明者不仅表明语句为真，还证明自己知道该statement成立的秘密证据。

老师讲的rewind是什么：

那如果有一个多项式时间的P，能说服V接受（P输出一个有效proof），那么存在一个多项式时间的extractor，它可以和P交互（也就是老师说的rewind P），最终把这个证明所依赖的witness挖出来。

而上面的Knowledge要求允许在安全证明里对P做rewind，以此来强调P必须真的掌握那个证据witness，否则就无法持续给出合格回答。

老师的例子里：

x是public input
w是secret witness

P发送给V：small size的π

V：检测π，输出1或0。

可以认为|T|≈|Circuit Size|。

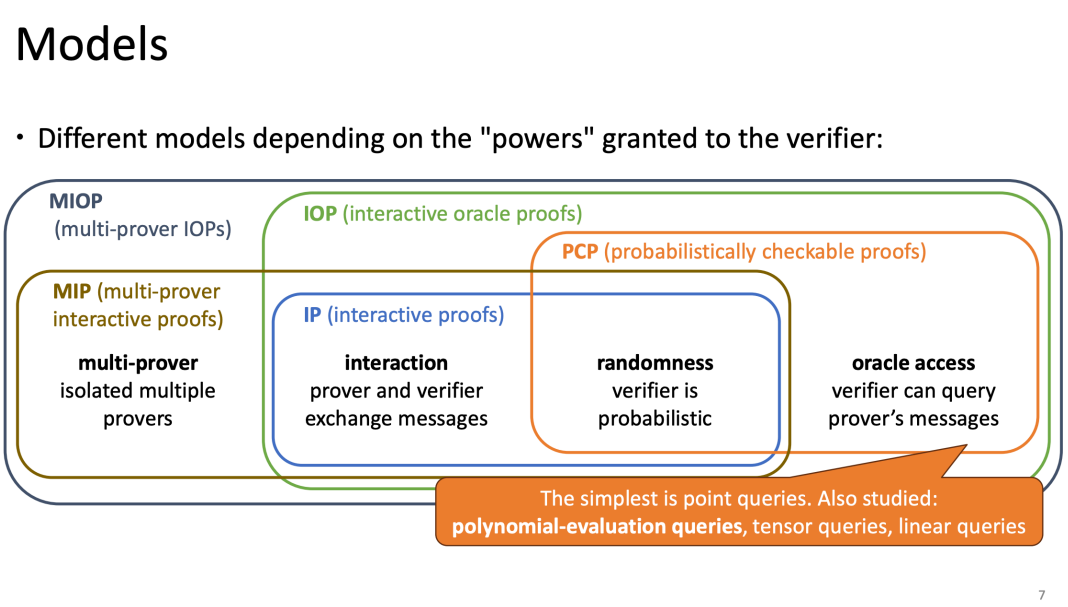
定义P的time bound：这个function可以在T time steps里完成证明。

定义V的time bound：由于V会把function和x都会读一遍，所以有poly(T) or poly(|F|+|x|+$\log T$)，其中T>|F|，T>|x|。

所有的snark都需要setup：Public（Transparent）setup和Private setup。
Public setup：setup的randomness完全是public的。
Private setup：setup的randomness不可以公开，只能setup时候用，setup后必须

丢弃（有毒废料）。

Universal setup：setup和function无关，只需要取到T的upper bound即可。对多个function可用。
Specific setup：与function强关联且绑定。构建出了最短的常数级别proof size，最少的 prover时间，常数验证时间。



PPT第7页

根据给V不同的能力来实现不同的证明系统：

oracle access：P发给V一个oracle（数组），V可以只读oracle的其中几位。

random access：希望V随机选择几位来读。

interaction：试图转化成非交互。

multi-prover：V和多个P交互，并且要求P之间独立，不可共谋（限制prover作恶的能力，如果能共谋那么多P和单P没区别）。

PCP：P发给V一个 oracle，V随机选读几个点，结束。

IP：V每次返回给P的挑战是随机的。

IOP：主流方向。P发送oracle，V读一些点，回复msg；重复。

MIP：多个P和V交互，V可以选择与哪个P继续交互，最终V得到0或1。只考虑MIP的话，V必须读P发送的整个msg，而不是读oracle。

MIOP：在MIP的基础上P发oracle。

不同oracle的种类可以设计不同的snark：

oracle也可以发polynomial：P发送f(x)给V，V不读整个f(x)，而是问oracle一些evaluation。

以及vector oracle等

Models

- Qualitative features:
  - IP: primarily sub-routines (e.g. sumcheck) to other probabilistic proofs.
  - PCP: pedagogically useful but mostly inefficient (e.g. with point queries).
  - MIP (& MIOP): attractive features (e.g. space efficiency) but hard to use.
  - IOP: underlie most efficient SNARKs.

8

PPT第8页

IP：V是要读P发送的整个msg的。一般用作整个protocol的Sub-routines。也就是说它不是完整的主协议，而是主流程里被反复调用的子协议模块。

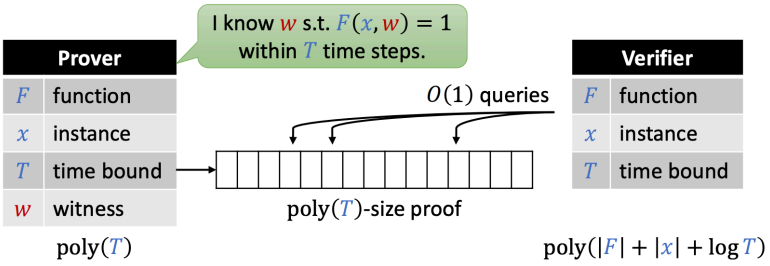
PCP:，只能发一个oracle。point queries PCPs不efficient，Linear queries PCPs是efficient。

MIP，MIOP：概念和性质上看起来很好，实际上很难用。那么如何变实用目前还是研究方向。

IOP：用的最多，可以多次发oracle。

Probabilistically Checkable Proofs

- The verifier is **probabilistic** and has **oracle access** to **1** prover message.



Note: PCP ≠ Succinct Argument!
It is insecure for the verifier to just ask the prover to answer a few queries.

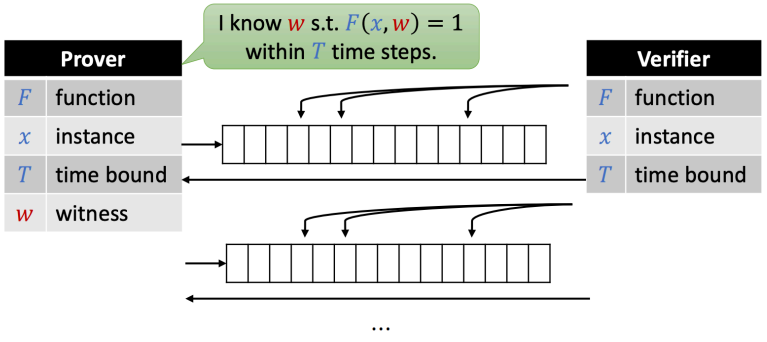
9

PPT第9页

Probabilistic：PCP不能interactive，只能发一个oracle。V可以生成随机挑战进行query。
V不读整个oracle size，得到O($\log T$)复杂度。但oracle是理想的，要求P发出去之后不能再更改。

Interactive Oracle Proofs

- The verifier can simultaneously leverage **randomness**, ***interaction***, and **oracle access**.



11

PPT第11页

PCP不好怎么办？用IOP。
流程：P发送oracle，V query，oracle返回值；重复步骤，最终V输出0或1。

如何构建SNARK？1.怎么去构建一个IOP，2.把IOP里的理想oracle转化成实际中可以用的（比如通过commitment）。那么focus IOP设计即可。
如果发送的是polynoimal IOP，那么就需要PCS。

Constructions of IOPs

- Flurry of IOP research in the past few years:
  - quasilinear-time ZK [BCGV16][BCFGRS17].
  - linear-size proof length [BCGRS17][RR20].
  - linear-time prover [BCGGHJ17][BCG20][BCL20].
  - linear-time proximity proofs [BBGR16][BBHR18][BKS18][BGKS20][BCIKS20][BN20].
  - efficient implementations [BBC+16][BBHR19][BCRSVW19][COS20].
- Many new techniques:
  - Interactive proof composition.
  - Univariate sumcheck.
  - Out-of-domain sampling.
  - Algebraic linking.

IOPs offer much improved efficiency (asymptotically & concretely).

12

PPT第12页

Interactive proof composition：比如有两个prove system，第一个proof size小，verifier time大，第二个相反。那么通过将两个系统结合，得到两个system的优点。

Realizing Proof Models: Cryptography

- Examples of SNARK recipes:

Probabilistic Proof	Cryptography	SNARK
linear PCP (and 2-message linear IP)	**linear encoding**	[G10][L11][BCIOP13] [GGPR13][PGHR13] [G16][GM17]…
PCP and IOP	**vector commitment**	Ligero, Aurora, Fractal, SCI, STARK, …
Polynomial PCP & IOP	**polynomial commitment**	Sonic, Marlin, Plonk, Spartan Supersonic-RSA, Hyrax, vSQL, vRAM, Libra, …

↓ type of computation (e.g., circuit vs machine)

↓
- cryptographic costs (in prover and in verifier)
- pre-quantum or post-quantum
- setup (public or private, specific or universal)

13

PPT第13页

对于IOP的oracle进行选择，再选择对应的commitment或者encoding等来构建query oracle。

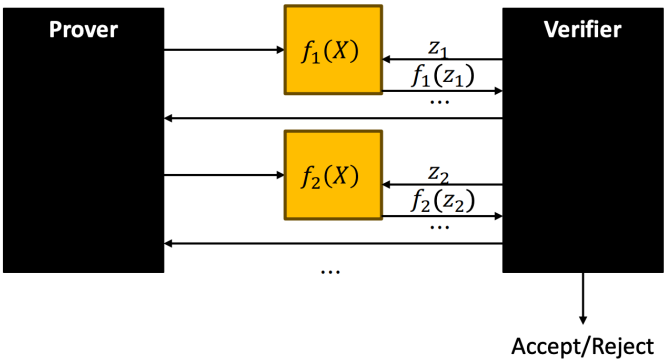
Probabilistic Proof列决定了计算的类型：

Circuit computation：比如把function的各种谓词和逻辑条件展开成Circuit，证明该Circuit是满足约束的。

Machine computation：把machine每一步的状态模拟，证明状态1到状态2，AIR和ZKVM那种感觉。

Cryptography列决定了P和V的cost，是否前量子or后量子，setup的种类。

Polynomial IOP



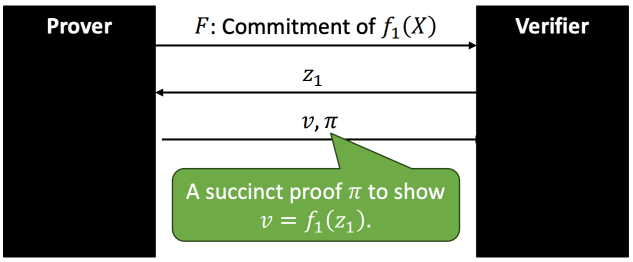
It can also support multi-variant polynomials.

14

PPT第14页

Polynomial IOP：每次P发的oracle是polynomial，V query 这些polynomial的evaluation；多次重复，最终输出1 or 0。

PCS Relation



$$\mathfrak{R} = \left\{ \begin{matrix} \text{witness: } f_1(X) \\ \text{public input: } F, z_1, v \end{matrix} \middle| \begin{matrix} F = Com(f_1) \\ v = f_1(z_1) \end{matrix} \right\}.$$

Can you write the relation for multivariant polynomial PCS?

15

PPT第15页

用PCS：对polynomial $f_1(X)$承诺，比如对它的系数向量算commitment得到 $\mathrm{Com}(f_1)$。

V返回evaluation query z_1，P返回$v = f_1(z_1)$，以及small size的π来证明$f_1(X)$在z_1处确实等于v。

其中，witness是$f_1(X)$，public input是$\mathsf{Com}(f_1)$，z_1和evaluation值v。

对于多变量PCS：把上述符号换成下面这些即可。

$f(\mathbf{X}) = f(X_1, \ldots, X_m) \in \mathbb{F}[\mathbf{X}]$，$\mathbf{z} = (z_1, \ldots, z_m) \in \mathbb{F}^m$ ，$v = f(\mathbf{z}) \in \mathbb{F}$.