NEW NON-INTERACTIVE ZERO-KNOWLEDGE SUBSET SUM, DECISION KNAPSACK AND RANGE ARGUMENTS

Helger Lipmaa

Bingsheng Zhang

University of Tartu and SUNY Buffalo

Zero-Knowledge

Needed always when participants are malicious

Insert ZK proofs

I know the meaning of life

… but I do not want to reveal it

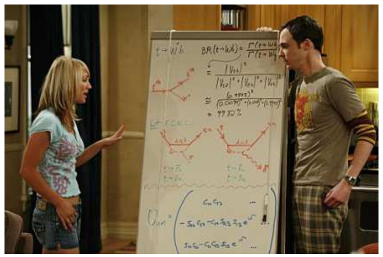
Interactive ZK

Blabla

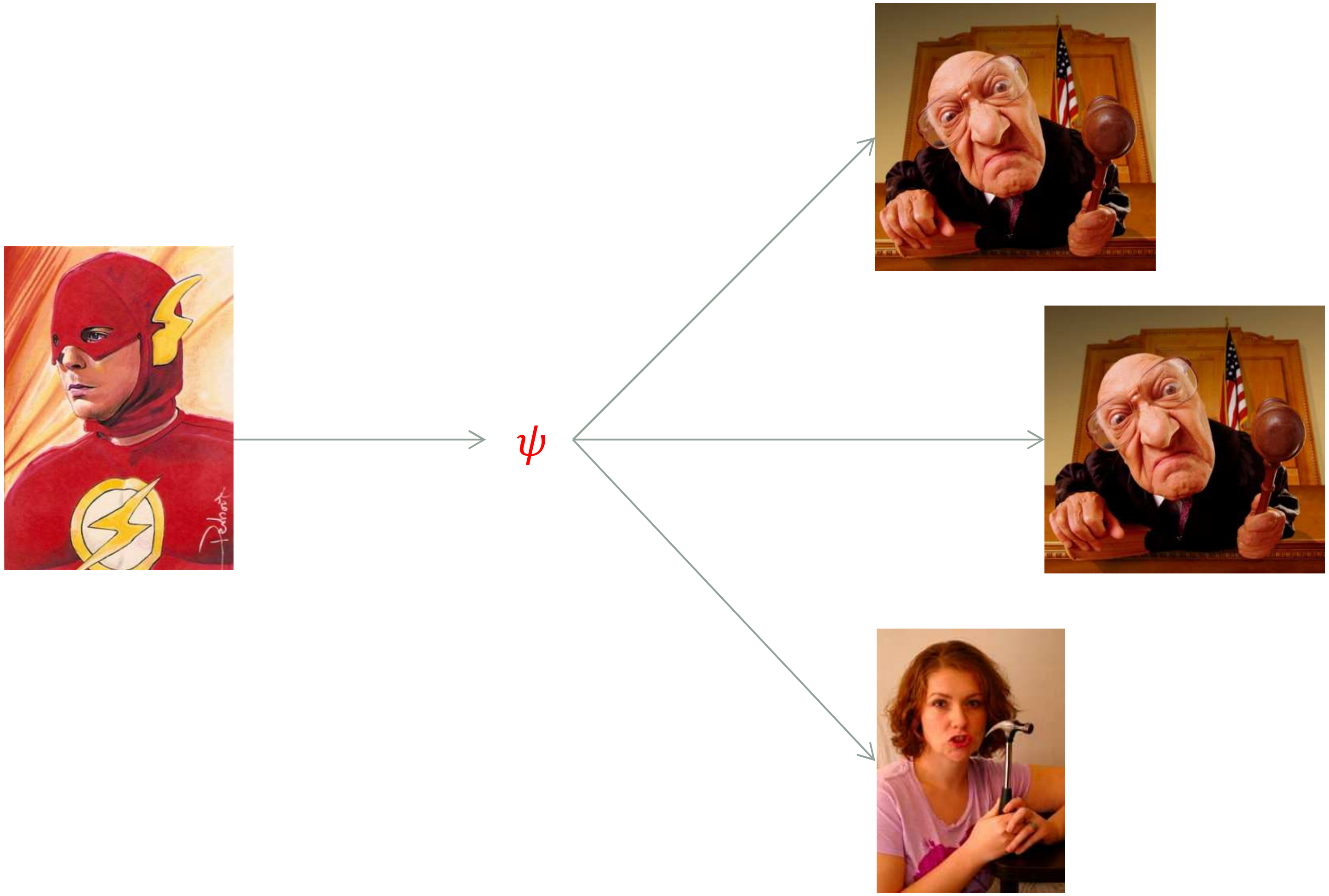
Challenge

…

Problem: need to interact every time



Non-Interactive ZK



NIZK: Requirements

witness

input

ψ

Correctness

Soundness

Zero Knowledge ← Proven by simulation

No NIZK in "standard model"

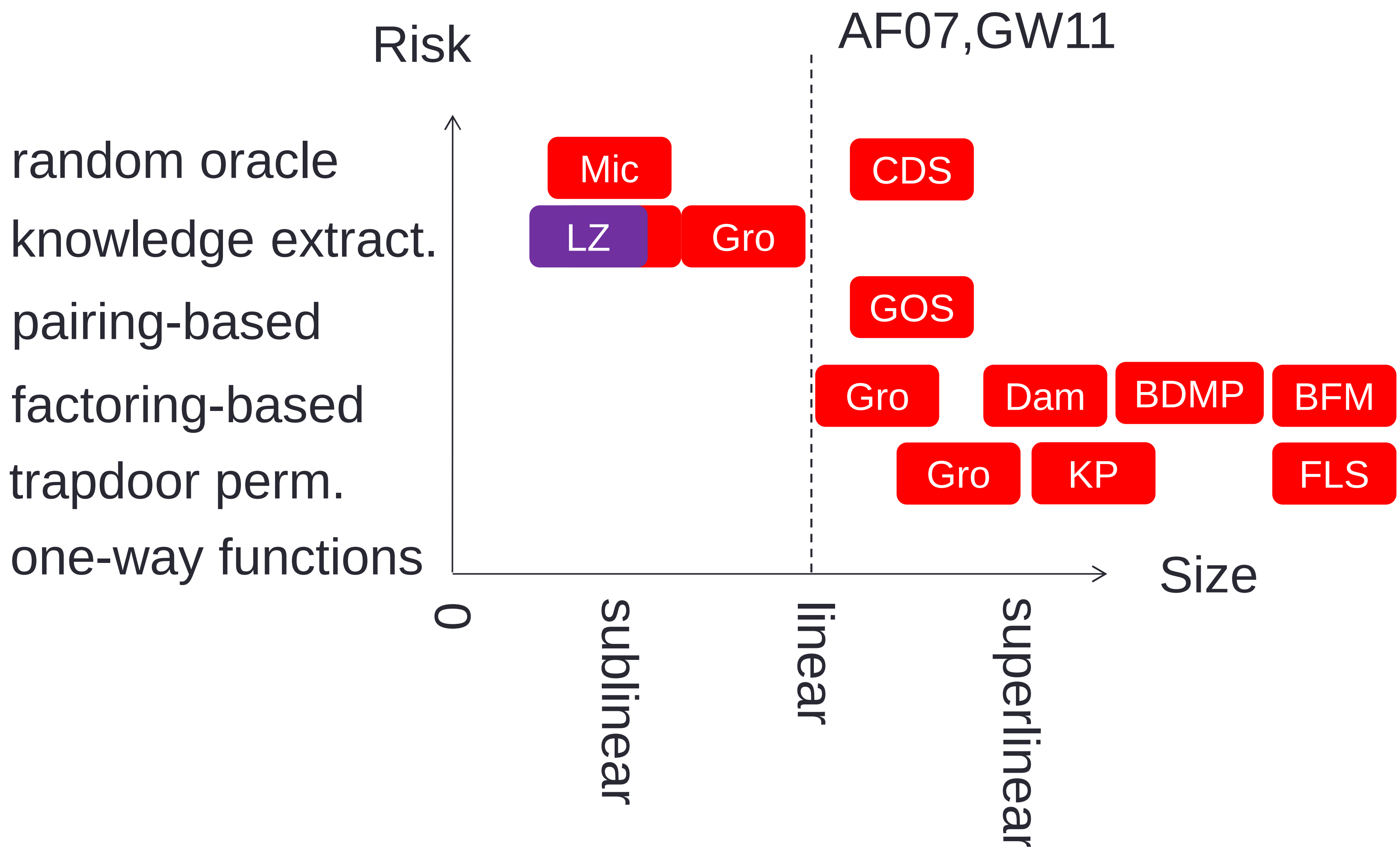
- Need simulator who can simulate conversation without knowing witness
- Simulator must have some extra power

CRS Model

- Parties have access to honestly generated common reference string
- In simulation, simulator can generate CRS together with trapdoor
- Does not rely on random oracles 
- [Abe Fehr 2007, Gentry Wichs 2011]:
  - nonstandard assumptions needed to get either non-interactive perfect zero-knowledge and sublinear communication
- **Here:** Knowledge assumptions, computational soundness

Size vs. assumption

{Invited talk, Jens Groth, TCC 2012}



Our Results: NIZK for Subset Sum

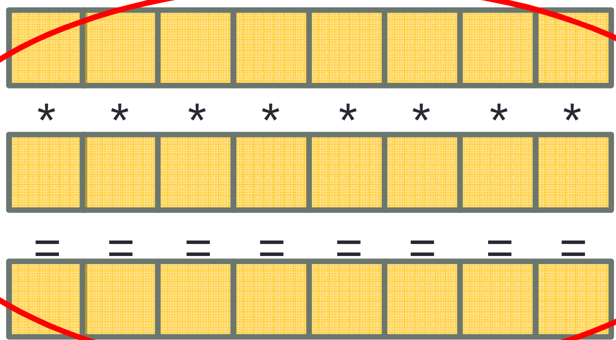
	Lang.	CRS length	Com.	Prover's comp.	Verifier's comp.
Groth 2010	CSAT	$\Theta(\vert C\vert^2)G$	$\Theta(1)G$	$\Theta(\vert C\vert^2)E$	$\Theta(\vert C\vert)E + \Theta(1)P$
Lipmaa 2012	CSAT	$\Theta(\vert C\vert^{1+o(1)})G$	$\Theta(1)G$	$\Theta(\vert C\vert^2)A$	$\Theta(\vert C\vert)E + \Theta(1)P$
This paper	SS	$\Theta(\vert S\vert^{1+o(1)})G$	$\Theta(1)G$	$\boldsymbol{\Theta(\vert S\vert^{1+o(1)})M}$	$\boldsymbol{\Theta(\vert S\vert)M + \Theta(1)P}$
		One	Many	One	Many

Why NIZK for NPC?

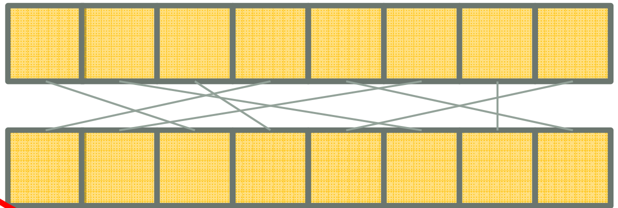
- Efficient NIZK for NPC L => efficient NIZK for all NP languages
  - By reduction
  - However, reduction “polynomial time” => usually not good enough
- Developed techniques are useful for other problems
  - True in our case ☺
  - [Chaabouni Lipmaa Zhang, FC 2012]: range proof
  - [Lipmaa Zhang, SCN 2012] : shuffle
  - Current results can be used to speed up CLZ12

Basic Arguments [Gro10, Lip12]

- Hadamard Product argument



- Permutation argument



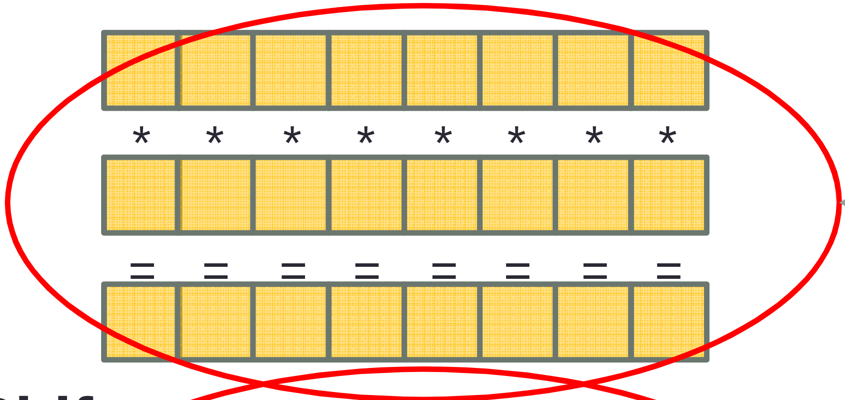
Quadratic (Groth) CRS or
Quasilinear (Lipmaa) CRS
Quadratic prover’s comp

Any permutation

- Parallel machine model

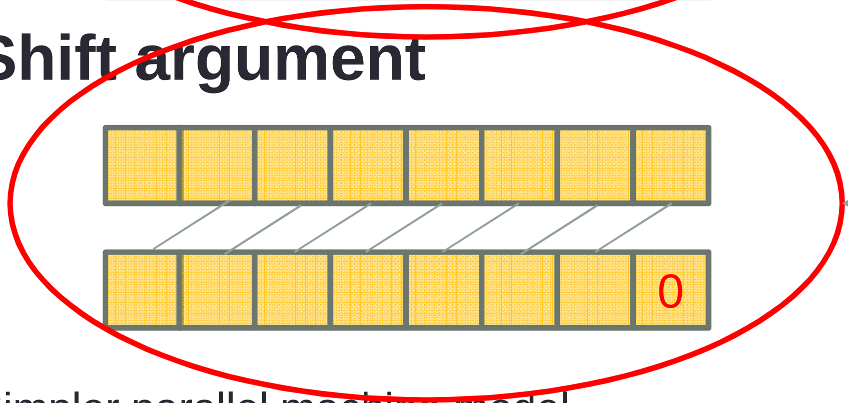
Simpler Basic Arguments

- Hadamard Product argument



Quasilinear CRS
Quasilinear comp

- **Shift argument**



Linear CRS
Linear prover's comp

- Simpler parallel machine model

Subset sum (in $\mathbb{Z}_p$)

- **Common input:** set $S = (s_1, \dots, s_n) \in \mathbb{Z}_p$
- **Task:** prove you *know* $\emptyset \neq T \subseteq S$, such that $\sum_{i \in T} i = 0$
  - ZK proof
  - Task is to *verify* subset sum, not to compute!
- Let $\vec{t}$ be the characteristic vector of T
  - $t_i = 1$ if $s_i \in T$

8371	-4038	-2347	2409	6383	-9350	2202	2424	-5372	-6074	-1734	-9046	1665	7372	-8846	4254	8856	9464	-1014	2301
-4188	-1500	-7809	-2721	-3884	472	4263	-3853	9414	-5087	2401	1315	-3740	-6484	3016	3463	-2530	-4317	2121	5026
7213	2355	3987	-4333	517	-5295	-9382	7425	8547	-4014	7061	9133	-4659	-1342	-7941	847	-9955	-2894	4987	-1417
9362	4945	6434	5789	-9889	-1915	-242	-1233	-9189	3378	6729	-2588	-6210	9630	-4789	-7958	9793	1672	5016	-3979
-9962	-1294	-1090	-4520	250	6493	3642	-748	8958	837	-2853	-6904	1461	-1833	1436	-1951	-1538	8651	-858	5062
8199	-7428	7068	-4436	6939	-6665	1854	-2707	-5748	-6982	4052	-2397	6093	1095	126	7455	3377	-3769	-1778	2552
5837	365	1557	5338	6465	2261	-6882	-9160	8655	6341	7281	-1910	1227	-7074	4688	-1116	-6024	-6038	-9118	-4849
-6057	-394	-9749	9565	7433	-1512	3728	-9656	8867	-8011	3028	-1777	3079	-5627	-7072	-2982	-8743	4113	-1448	-9305
-8599	-3894	8865	2350	-6878	-5015	3027	-6362	2548	-1795	-422	-5029	-7282	2874	-9765	6472	7588	4879	-3901	6895
-1465	4807	6974	-2936	-2817	1752	-8008	-6082	-798	-8426	-4815	2590	1971	-9881	8328	-1697	-4967	2688	5385	495
8211	-7758	8012	7873	3647	-8325	4763	7360	4632	7605	8533	4999	4893	-9397	-4936	-9284	8685	2090	-8036	-342
7817	9358	6254	-6976	-6744	3786	5993	7156	-9102	-2283	-313	2499	9008	5	3590	-4593	-1785	-8249	-4771	-7350
9417	-7032	-5966	5758	-901	-1824	-4339	3577	-8483	-8515	-4603	939	2867	-9518	5846	-6281	7862	-8888	-9873	6296
5618	4951	-916	-3927	-5954	7129	-456	5022	8755	6476	3323	-3766	-4678	-1787	-1063	-2808	9351	3623	545	-8725
2743	-2401	-7459	-4607	-1544	5725	4790	-6397	-2860	-6833	-1335	2149	1369	5302	-581	-4609	-9927	-9494	-321	-6538
6239	-4556	9340	-3327	3421	-582	8604	-1999	5807	4886	6486	-2762	-4116	2620	5541	-9683	-7822	717	1729	3008
7183	-6223	-2520	-984	8671	-5994	-4441	1651	-5716	-8568	7331	-3662	-7415	2244	4538	228	-5445	-8335	5426	8277
8731	-6390	3281	7650	-2057	-9692	-7137	-7712	214	8123	7790	4330	-9612	-8250	-9561	8542	8915	-6937	9161	-3599
-6800	-6680	-7883	-4454	-3063	-3283	-9421	7700	1851	8506	6428	2295	-7810	-6365	-2998	9314	8023	-4928	750	-5801
9202	2071	-3255	2577	530	-7578	-9191	-5812	-9491	7554	-6678	9083	5464	697	5916	-5777	-5297	8078	3061	-1046

Solution found. The selected subset has a sum of zero.

Randomize Set | Find Subset

Subset sum: Argument idea

- Let $\vec{b} = \vec{s} \circ \vec{t}$ /* $b_i = s_i$ if $s_i \in T$, $b_i = 0$ otherwise */
- Let $\vec{c}$ be such that $c_i = \sum_{j \geq i} b_j$, $\vec{d}$ be shift of $\vec{c}$, $d_i = c_{i+1}$
- Commit to $\vec{t}$, $\vec{b}$, $\vec{c}$, $\vec{d}$
- Prove in ZK that $\vec{t}$ is Boolean /* $\vec{t} \circ \vec{t} = \vec{t}$, product argument */
- Prove in ZK that $\vec{t}$ is non-zero /* efficient */
- Prove in ZK that $\vec{s} \circ \vec{t} = \vec{b}$ /* product argument */
- Prove in ZK that $c_i = \sum_{j \geq i} b_j$
  - Prove in ZK that $\vec{d}$ is a shift of $\vec{c}$ /* $d_n = 0$, $d_i = c_{i+1}$ */
  - Check that $\vec{c} = \vec{b} + \vec{d}$ /* $c_n = b_n$, $c_i = b_i + c_{i+1} = \sum_{j \geq i} b_j$, $c_1 = \sum_j b_j$ */
- Prove in ZK that $c_1 = 0$ /* easy */

Product Argument

- Given commitments to $\vec{a}$, $\vec{b}$, $\vec{c}$, prove in ZK that $\{c_i = a_i b_i\}$
- Based on [Lipmaa 2012]
  - Uses a progression-free set $\Lambda = (\lambda_1, \dots, \lambda_n)$, $\lambda_n = o(n 2^{2\sqrt{2 \log_2 n}})$
  - Most expensive part in computation:
    - $\prod_{i=1}^{n} \prod_{j \neq i} g^{(a_i b_j - c_i) x^{\lambda_i + \lambda_j}}$, given $\{g^{x^k} : k \in \{\lambda_i + \lambda_j : i \neq j\}\}$
  - [Lipmaa 2012]: can do in $\cancel{n^2}$ additions and $o(n 2^{2\sqrt{2 \log_2 n}})$ ~~exp~~.
- [This paper]:

  Does not work with permutation argument

  - Use Fast Fourier Transform to compute all exponents in $o\left(n 2^{2\sqrt{2 \log_2 n}} \log n\right) \ll n^2$ multiplications in $\mathbb{Z}_p$
  - Use Pippenger's algorithm to compute multi-exponentiation by doing $o(n 2^{2\sqrt{2 \log_2 n}})$ multiplications in elliptic curve group

Shift Argument: Preliminaries

- Let
  - e be bilinear map, $e\left(g^a, g^b\right) = e(g, g)^{ab}$
  - $\Lambda = (\lambda_1, \dots, \lambda_n)$ be a progression-free set, $\Lambda \subset \{1, \dots, N\}$ for $N \ll n^2$
  - $\upsilon > \lambda_n$ be a large integer
  - σ be a secret key
  - $g^{\sigma^\upsilon}, \{g^{\sigma^{\lambda_i}}\}$ are given in CRS
  - $Com(\vec{a}; r) := \left(g^{\sigma^\upsilon}\right)^r \cdot \prod_{i=1}^n \left(g^{\sigma^{\lambda_i}}\right)^{a_i}$
- Note that $\log_g Com(\vec{a}; r) = r\sigma^\upsilon + \sum_{i=1}^n a_i \sigma^{\lambda_i}$

In previous papers, $\upsilon = 0$

Shift Argument: Brief Idea

- Let $A = Com(\vec{a}; r_a)$ and $B = Com(\vec{b}; r_b)$
- Consider "verification equation" $e(A, g^{\sigma})/e(B, g) = e(g, \pi)$
- After taking discrete logarithm of left side we get
- $\left(r_a\sigma^{v} + \sum_{i=1}^{n} a_i\sigma^{\lambda_i}\right)\sigma - \left(r_b\sigma^{v} + \sum_{i=1}^{n} b_i\sigma^{\lambda_i}\right) =$
- $\cancel{\sum_{i=2}^{n}\underbrace{(a_{i-1} - b_i)}\sigma^{\lambda_{i-1}+1} + \underbrace{a_n}\sigma^{\lambda_n+1}} + F_{\pi}(\sigma)$

0, if the prover is honest

$F_{\pi}(X) = \sum_{\phi\in\Phi} f_{\phi}\phi(X)$

$X^{\lambda_{i-1}+1}, X^{\lambda_n+1} \notin \operatorname{span}\Phi$

- Prover proves he an represent $\log\pi$ as $F_{\pi}(\sigma)$ (for *some* coefficients f_{ϕ})

Conclusions

- NIZK proof for NP-complete language subset sum
  - Based on two basic arguments, product and shift
  - “NIZK programming language”
  - Slightly modified commitment scheme
- Product argument:
  - [Lipmaa 2012]: quadratic prover’s computation
  - This paper: quasilinear complexity by using FFT, Pippenger’s multi-exponentiation algorithm
- Shift argument:
  - Completely new, linear complexity
  - Replaces permutation argument (quadratic prover’s comp.)

Conclusions

- More efficient range argument:
  - [Chaabouni Lipmaa Zhang 2012]: replace permutation with shift
- Decision knapsack argument:
  - Combine subset sum argument with range argument
- [Lipmaa 2012] had Circuit-SAT argument with quadratic complexity --- we showed one can do other NPC languages with less work
- **Question:** how efficient direct NIZK one can build for different NPC languages?
  - In a concrete parallel machine model
  - … what are other nice basic arguments?