Towards Language Model Guided TLA$^+$ Proof Automation

Yuhao Zhou and Stavros Tripakis

Northeastern University

What is this talk about?

An **LLM-guided** method for

automatically generating

TLA$^+$ proofs

by **hierarchical decomposition** of

proof obligations

TLA$^+$ Proofs

TLA$^+$ Proofs and TLAPS

```
1  -------- MODULE sums_even -------
2  EXTENDS Naturals, TLAPS
3
4  Even(x) == x % 2 = 0
5
6  THEOREM L0 == ASSUME NEW x ∈ Nat PROVE
         Even(x + x) = Even(x * 2)
7  OBVIOUS
8
9  THEOREM L1 == ASSUME NEW x ∈ Nat PROVE
         Even(x * 2) = ((x * 2) % 2 = 0)
10 BY DEF Even
11
12 THEOREM T1 == ASSUME NEW x ∈ Nat PROVE
         Even(x + x)
13 BY L0, L1 DEF Even
14 ========================
```

TLA$^+$ Proofs and TLAPS

```
1  -------- MODULE sums_even -------
2  EXTENDS Naturals, TLAPS
3
4  Even(x) == x % 2 = 0
5
6  THEOREM L0 == ASSUME NEW x ∈ Nat PROVE
         Even(x + x) = Even(x * 2)
7  OBVIOUS
8
9  THEOREM L1 == ASSUME NEW x ∈ Nat PROVE
         Even(x * 2) = ((x * 2) % 2 = 0)
10 BY DEF Even
11
12 THEOREM T1 == ASSUME NEW x ∈ Nat PROVE
         Even(x + x)
13 BY L0, L1 DEF Even
14 ====================
```

Target Proof Obligation: $x + x$ is even

TLA$^+$ Proofs and TLAPS

```
1  -------- MODULE sums_even -------
2  EXTENDS Naturals, TLAPS
3
4  Even(x) == x % 2 = 0
5
6  THEOREM L0 == ASSUME NEW x ∈ Nat PROVE
       Even(x + x) = Even(x * 2)
7  OBVIOUS
8
9  THEOREM L1 == ASSUME NEW x ∈ Nat PROVE
       Even(x * 2) = ((x * 2) % 2 = 0)
10 BY DEF Even
11
12 THEOREM T1 == ASSUME NEW x ∈ Nat PROVE
       Even(x + x)
BY L0, L1 DEF Even
=====================
```

Use L0 and L1 to prove the target proof obligation

Target Proof Obligation: $x + x$ is even

TLA$^+$ Proofs and TLAPS

TLA$^+$ Proof System (TLAPS) will try to automatically discharge the proof obligation

```
1  -------- MODULE sums_even -------
2  EXTENDS Naturals, TLAPS
3
4  Even(x) == x % 2 = 0
5
6  THEOREM L0 == ASSUME NEW x ∈ Nat PROVE
       Even(x + x) = Even(x * 2)
7  OBVIOUS
8
9  THEOREM L1 == ASSUME NEW x ∈ Nat PROVE
       Even(x * 2) = ((x * 2) % 2 = 0)
10 BY DEF Even
11
12 THEOREM T1 == ASSUME NEW x ∈ Nat PROVE
       Even(x + x)
13 BY L0, L1 DEF Even
14 ====================
```

Use L0 and L1 to prove the target proof obligation

Target Proof Obligation: $x + x$ is even

TLA$^+$ Proofs vs Tactics-based Proofs

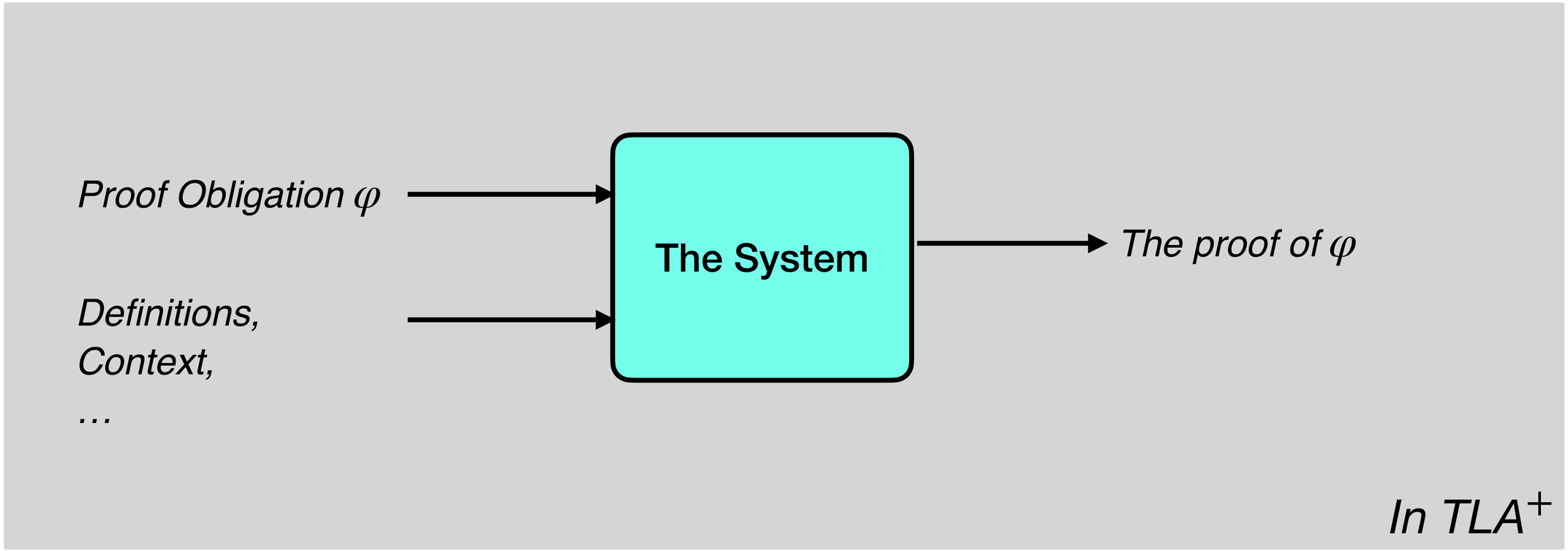
```
1  -------- MODULE sums_even -------
2  EXTENDS Naturals, TLAPS
3
4  Even(x) == x % 2 = 0
5
6  THEOREM L0 == ASSUME NEW x ∈ Nat PROVE
       Even(x + x) = Even(x * 2)
7  OBVIOUS
8
9  THEOREM L1 == ASSUME NEW x ∈ Nat PROVE
       Even(x * 2) = ((x * 2) % 2 = 0)
10 BY DEF Even
11
12 THEOREM T1 == ASSUME NEW x ∈ Nat PROVE
       Even(x + x)
13 BY L0, L1 DEF Even
14 ========================
```

```
1  import Mathlib.Tactic.Ring
2
3  def even (x : Nat) : Prop := x % 2 = 0
4
5  theorem T1 : ∀ x : Nat, even (x+x) := by
6      intro x
7      ring_nf
8      dsimp [even]
9      simp
```

The development of TLA$^+$ proofs focuses more on breaking down proof obligations instead of building a list of tactics.

Our Goal

Our Goal


Proof Obligation φ
Definitions,
Context,
…
The System
The proof of φ
In TLA$^+$

Methods

TLAPS OBVIOUS

The Naive Method 1

```
1  -------- MODULE sums_even -------
2  EXTENDS Naturals, TLAPS
3
4  Even(x) == x % 2 = 0
5
6  THEOREM L0 == ASSUME NEW x ∈ Nat PROVE
       Even(x + x) = Even(x * 2)
7  OBVIOUS
8
9  THEOREM L1 == ASSUME NEW x ∈ Nat PROVE
       Even(x * 2) = ((x * 2) % 2 = 0)
10 BY DEF Even
11
12 THEOREM T1 == ASSUME NEW x ∈ Nat PROVE
       Even(x + x)
13 BY L0, L1 DEF Even
14 ========================
```

TLAPS OBVIOUS

The Naive Method 1

```
1  -------- MODULE sums_even -------
2  EXTENDS Naturals, TLAPS
3
4  Even(x) == x % 2 = 0
5
6  THEOREM L0 == ASSUME NEW x ∈ Nat PROVE
         Even(x + x) = Even(x * 2)
7  OBVIOUS
8
9  THEOREM L1 == ASSUME NEW x ∈ Nat PROVE
         Even(x * 2) = ((x * 2) % 2 = 0)
10 BY DEF Even
11
12 THEOREM T1 == ASSUME NEW x ∈ Nat PROVE
         Even(x + x)
13 BY L0, L1 DEF Even
14 ========================
```

Just use OBVIOUS.

TLAPS OBVIOUS

The Naive Method 1

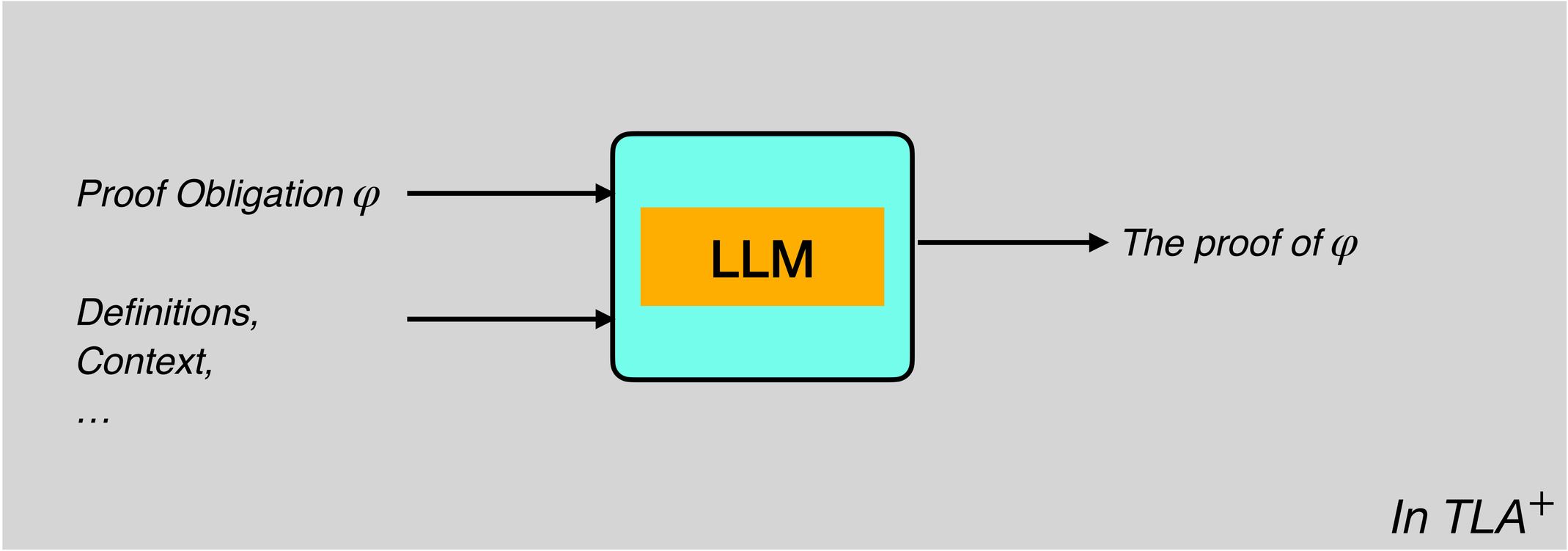
```
1  -------- MODULE sums_even -------
2  EXTENDS Naturals, TLAPS
3
4  Even(x) == x % 2 = 0
5
6  THEOREM L0 == ASSUME NEW x ∈ Nat PROVE
         Even(x + x) = Even(x * 2)
7  OBVIOUS
8
9  THEOREM L1 == ASSUME NEW x ∈ Nat PROVE
         Even(x * 2) = ((x * 2) % 2 = 0)
10 BY DEF Even
11
12 THEOREM T1 == ASSUME NEW x ∈ Nat PROVE
         Even(x + x)
13 BY L0, L1 DEF Even
14 ========================
```

Just use `OBVIOUS`.

- Cannot handle complex proof obligations.

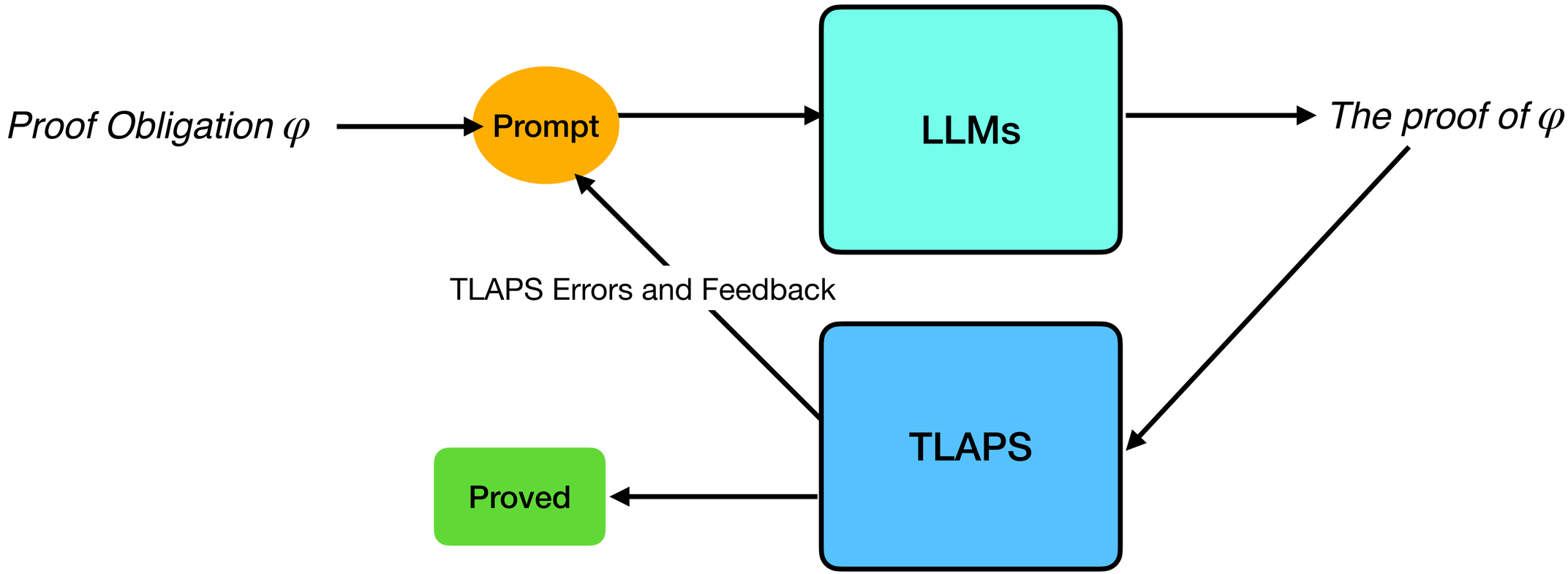
Direct LLM-Based Proof Generation (DLPG)

The Naive Method 2



Direct LLM-Based Proof Generation (DLPG)

The Naive Method 2



Direct LLM-Based Proof Generation (DLPG)

Issues

Direct LLM-Based Proof Generation (DLPG)

Issues

TLA$^+$, especially its proof language, is a low-resource language.

LLMs frequently generate proofs with incorrect syntax.

```
1  ---- MODULE amc12a_2002_p6 ----
2  EXTENDS Integers
3
4  THEOREM amc12a_2002_p6 ==
5    ∀ n ∈ Nat \ {0} :
6      ∃ m ∈ Nat :
7        (m > n) ∧ (∃ p ∈ Nat : m * p ≤ m +
       p)
8  PROOF
9  <1>1. FIX n ∈ Nat \ {0}.
10 <1>2. TAKE m = n + 1.
11 <1>3. HAVE m > n BY INT_ARITH.
12 <1>4. TAKE p = 1.
13 <1>5. HAVE m * p ≤ m + p BY INT_ARITH.
14 <1>6. QED
15 ====
```

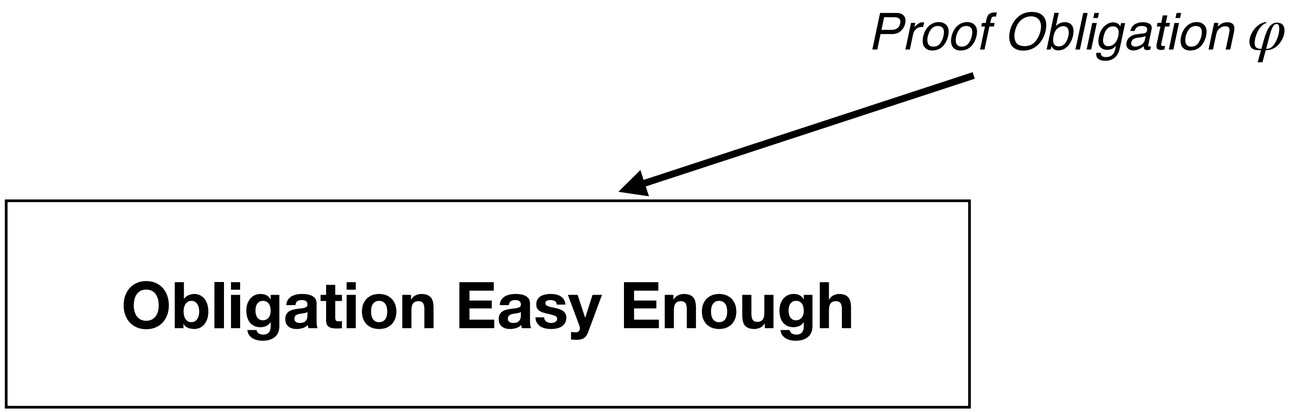
Two naive methods do not work.
One step back …

Proving an Obligation

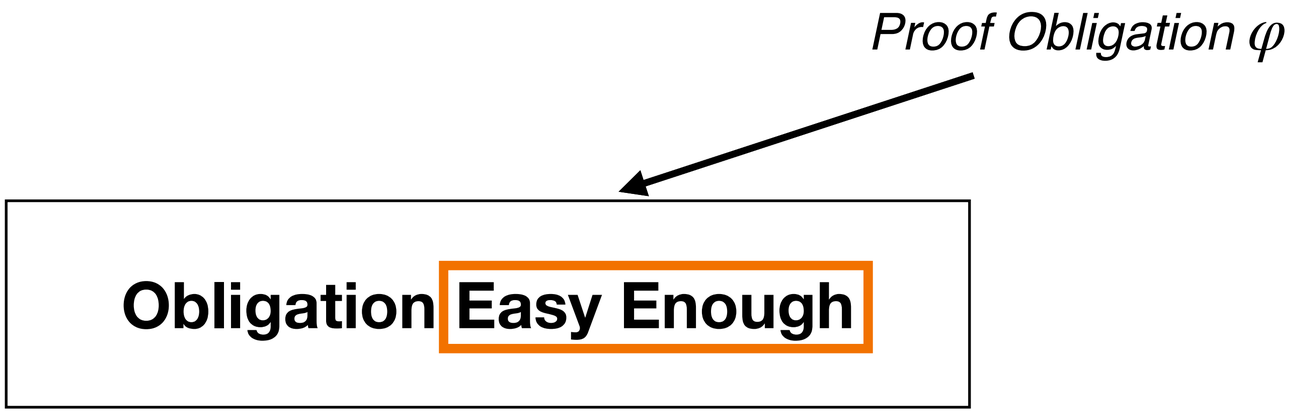
Proving an Obligation

Proof Obligation φ

Proving an Obligation

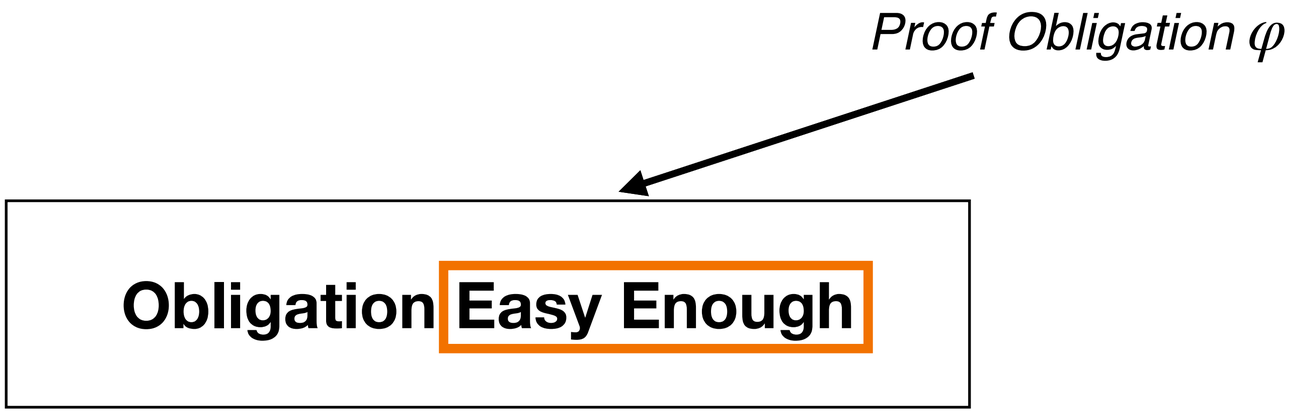


Proving an Obligation



Means that if all necessary definitions and assumptions are provided, TLAPS can prove the obligation

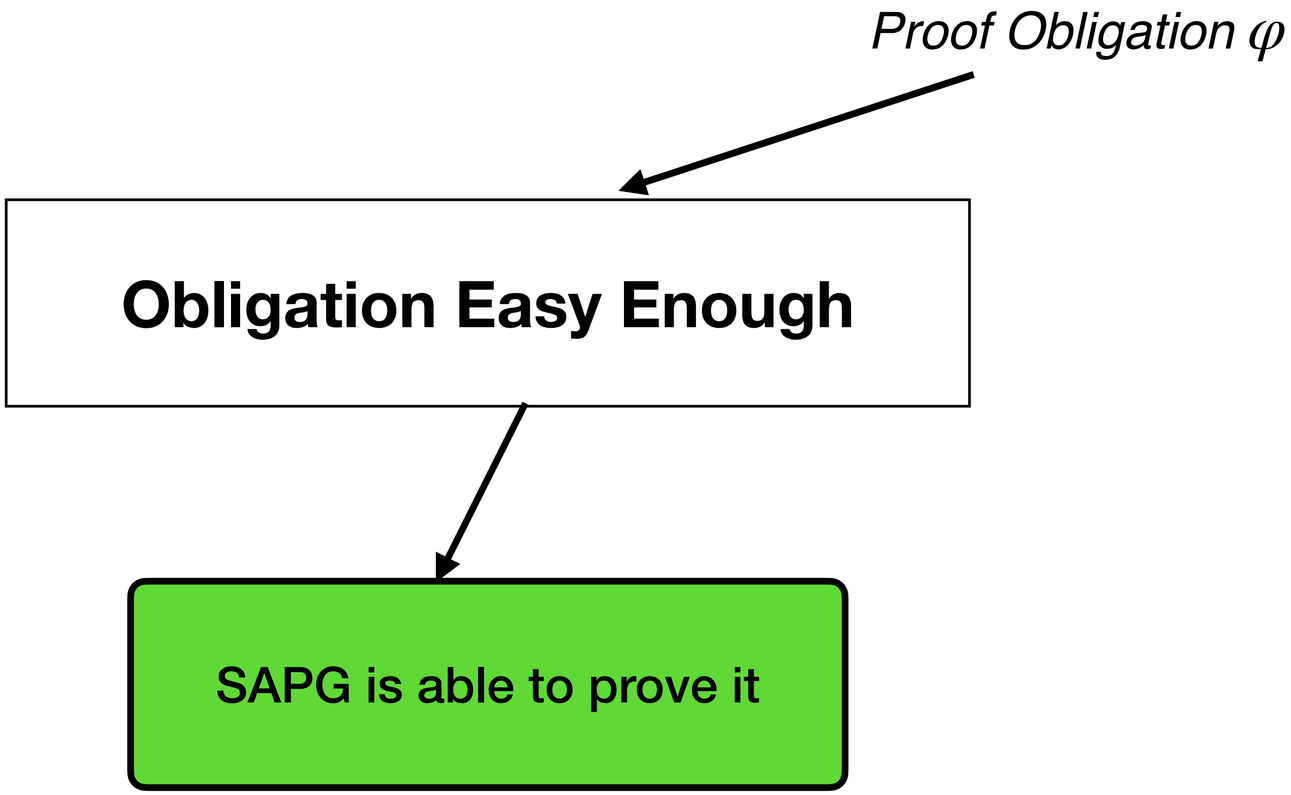
Proving an Obligation



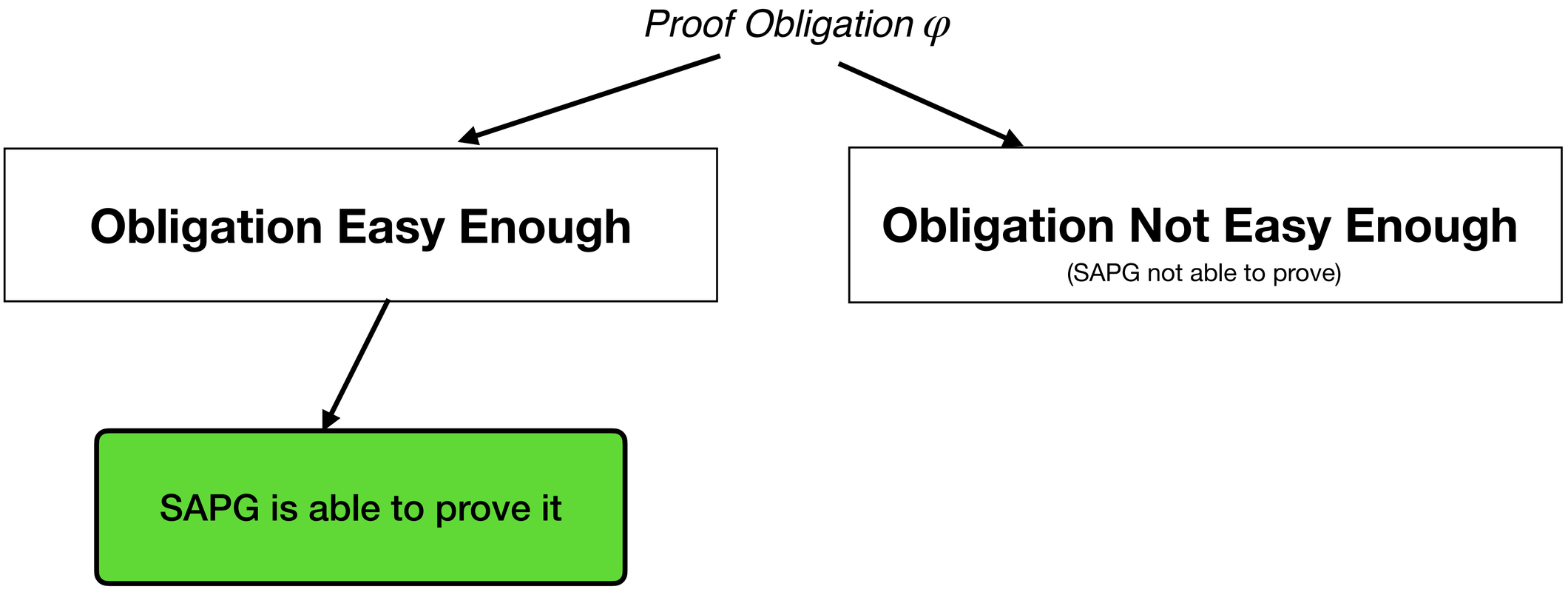
Means that if all necessary definitions and assumptions are provided, TLAPS can prove the obligation

We built a tool called **SAPG** that can analyze the syntax tree and collect all necessary assumptions and definitions, then construct the proof for φ

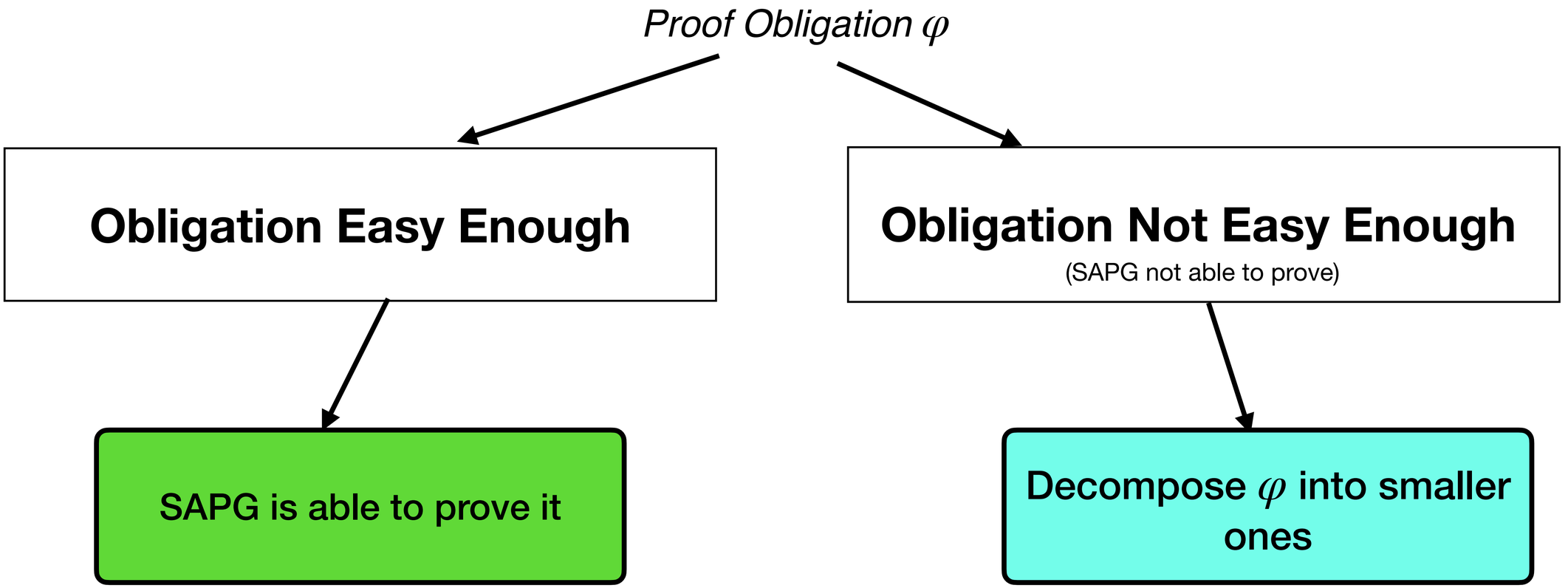
Proving an Obligation



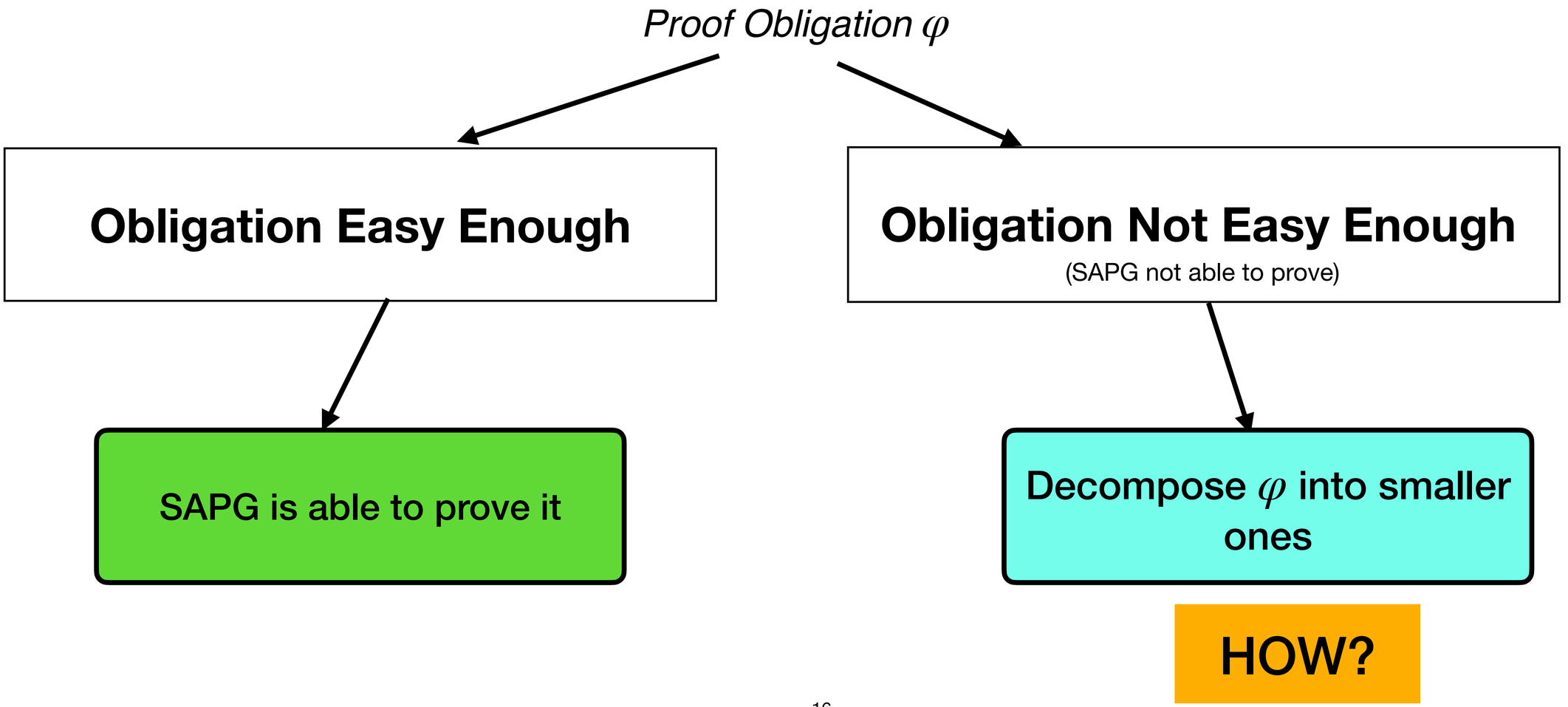
Proving an Obligation



Proving an Obligation



Proving an Obligation

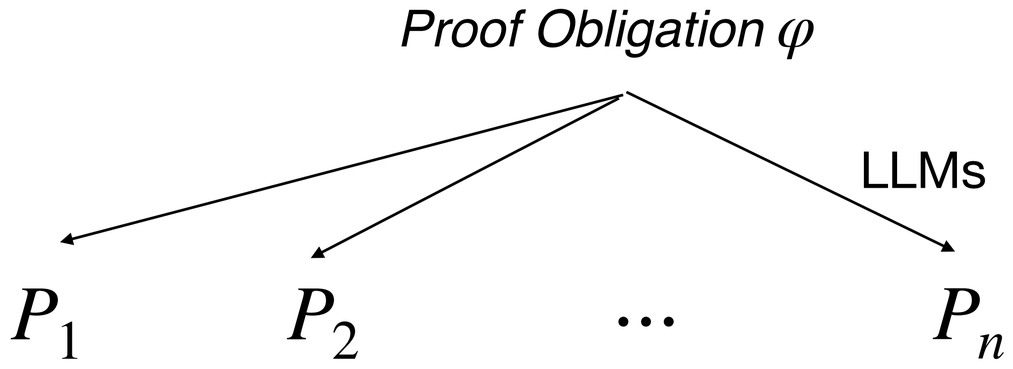


Language Model Guided Obligation Decomposition

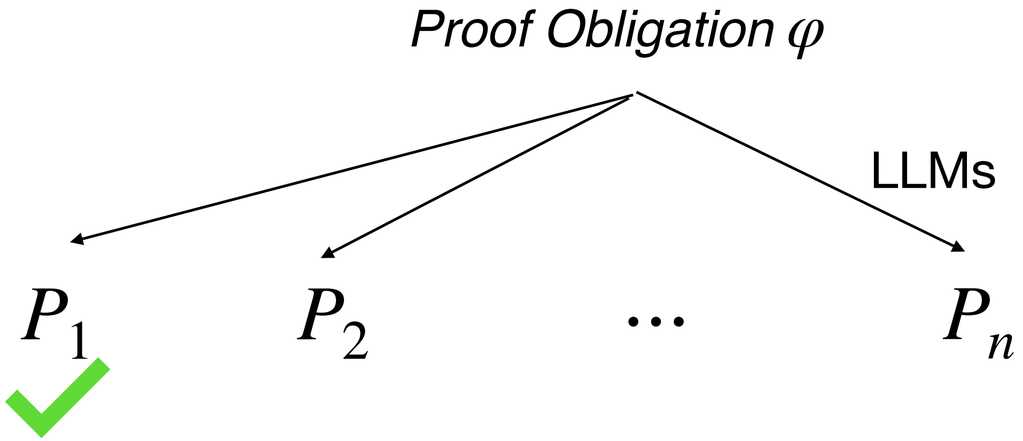
Language Model Guided Obligation Decomposition

Proof Obligation φ

Language Model Guided Obligation Decomposition



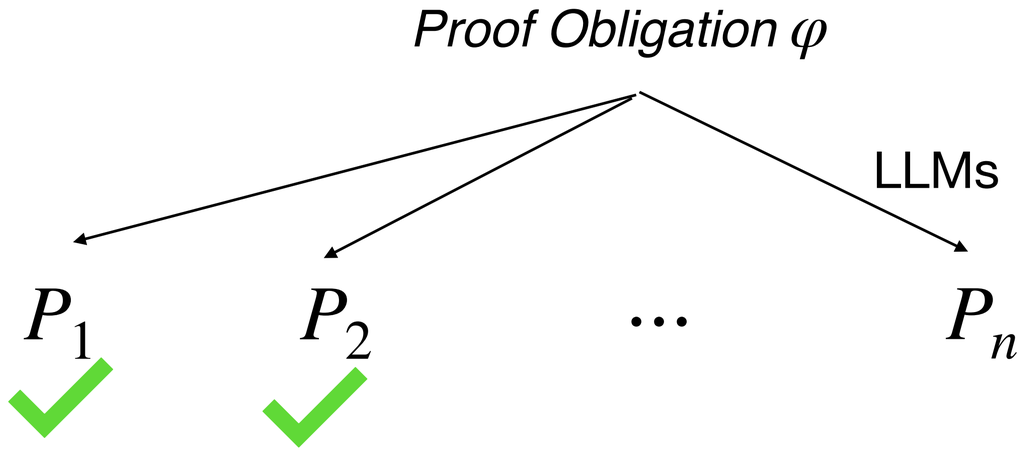
Language Model Guided Obligation Decomposition



✓ SAPG can prove it

✗ SAPG cannot prove it

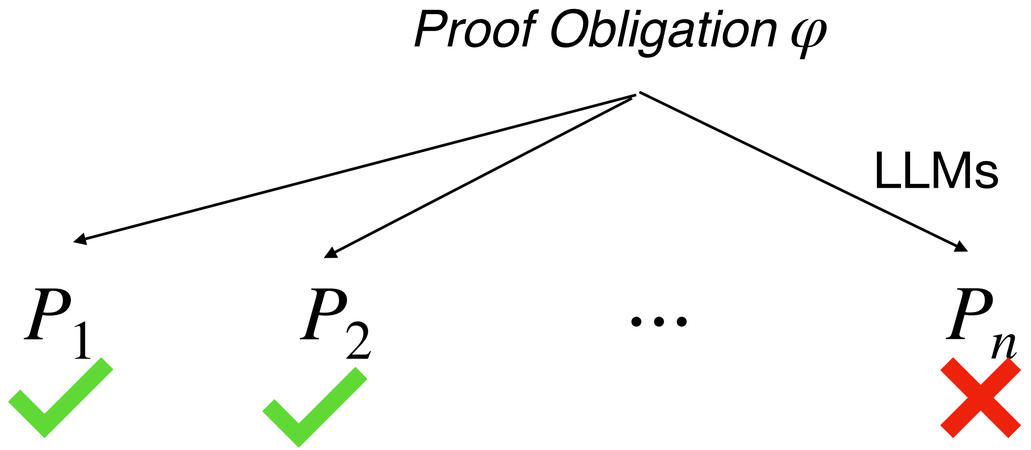
Language Model Guided Obligation Decomposition



✓ SAPG can prove it

✗ SAPG cannot prove it

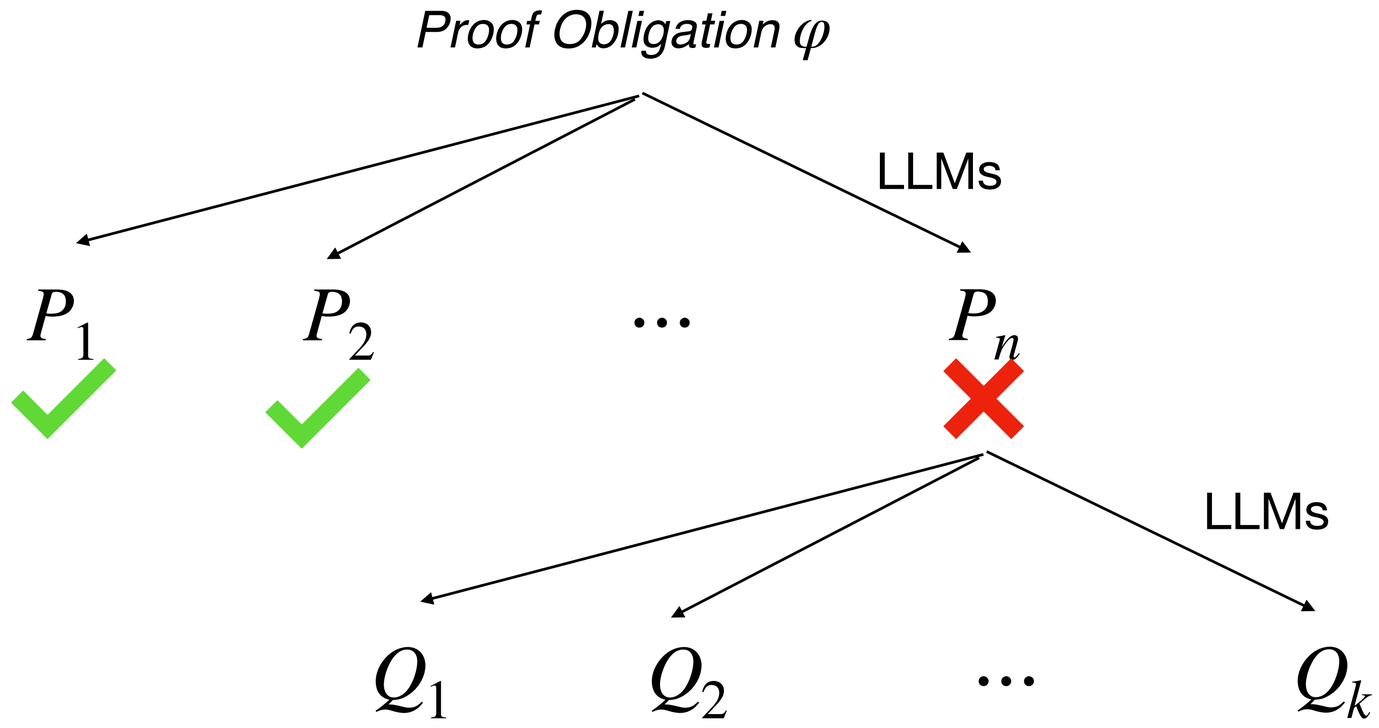
Language Model Guided Obligation Decomposition



✓ SAPG can prove it

✗ SAPG cannot prove it

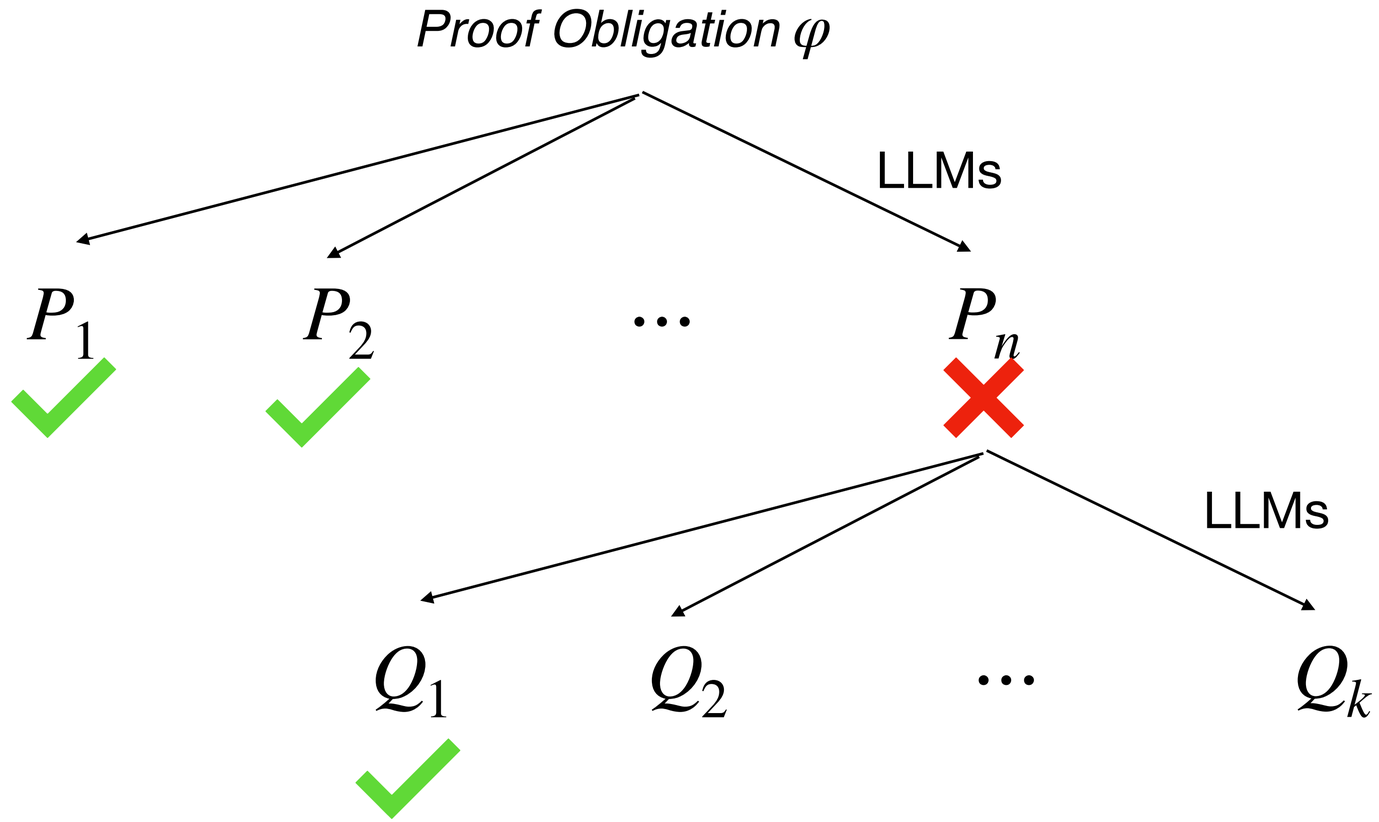
Language Model Guided Obligation Decomposition



SAPG can prove it

SAPG cannot prove it

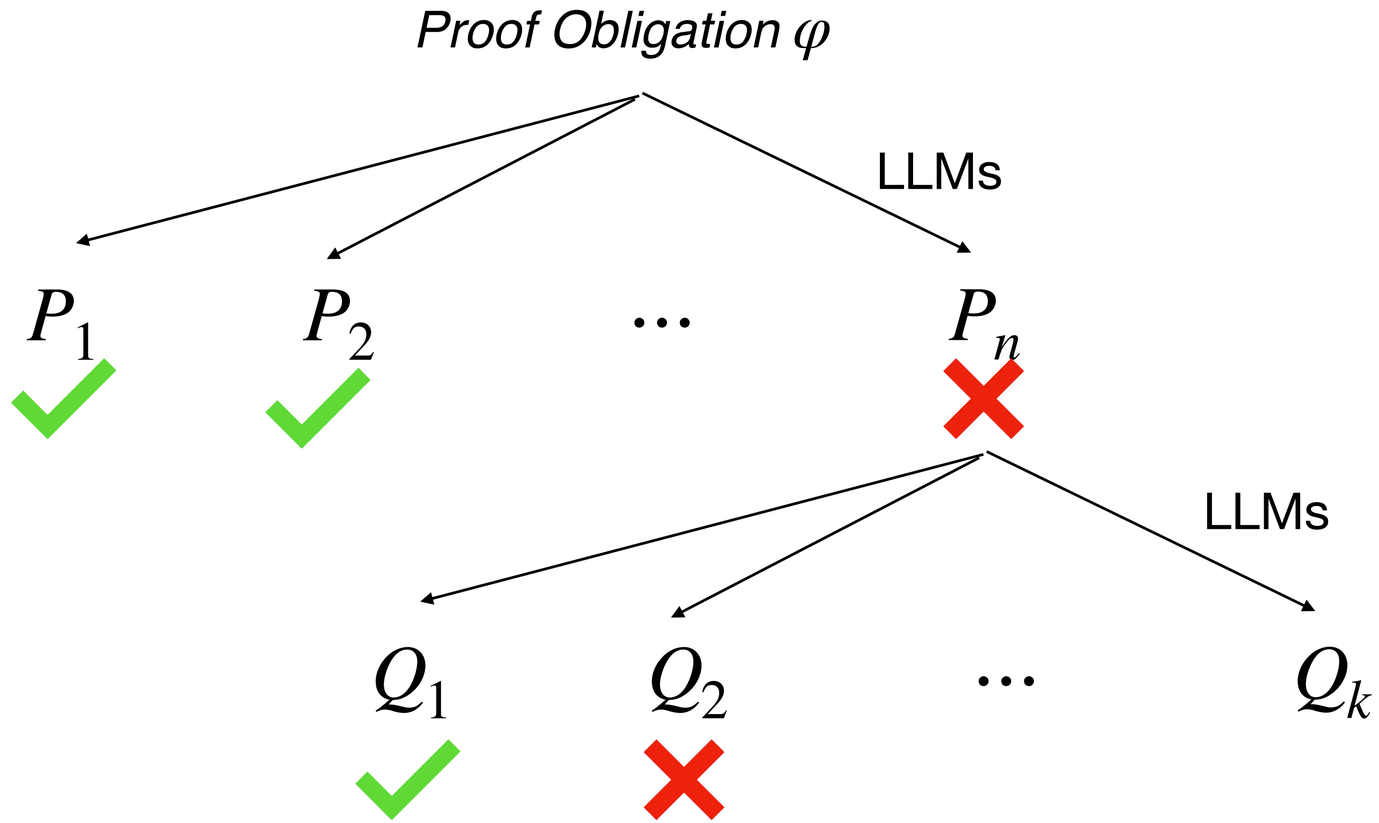
Language Model Guided Obligation Decomposition



SAPG can prove it

SAPG cannot prove it

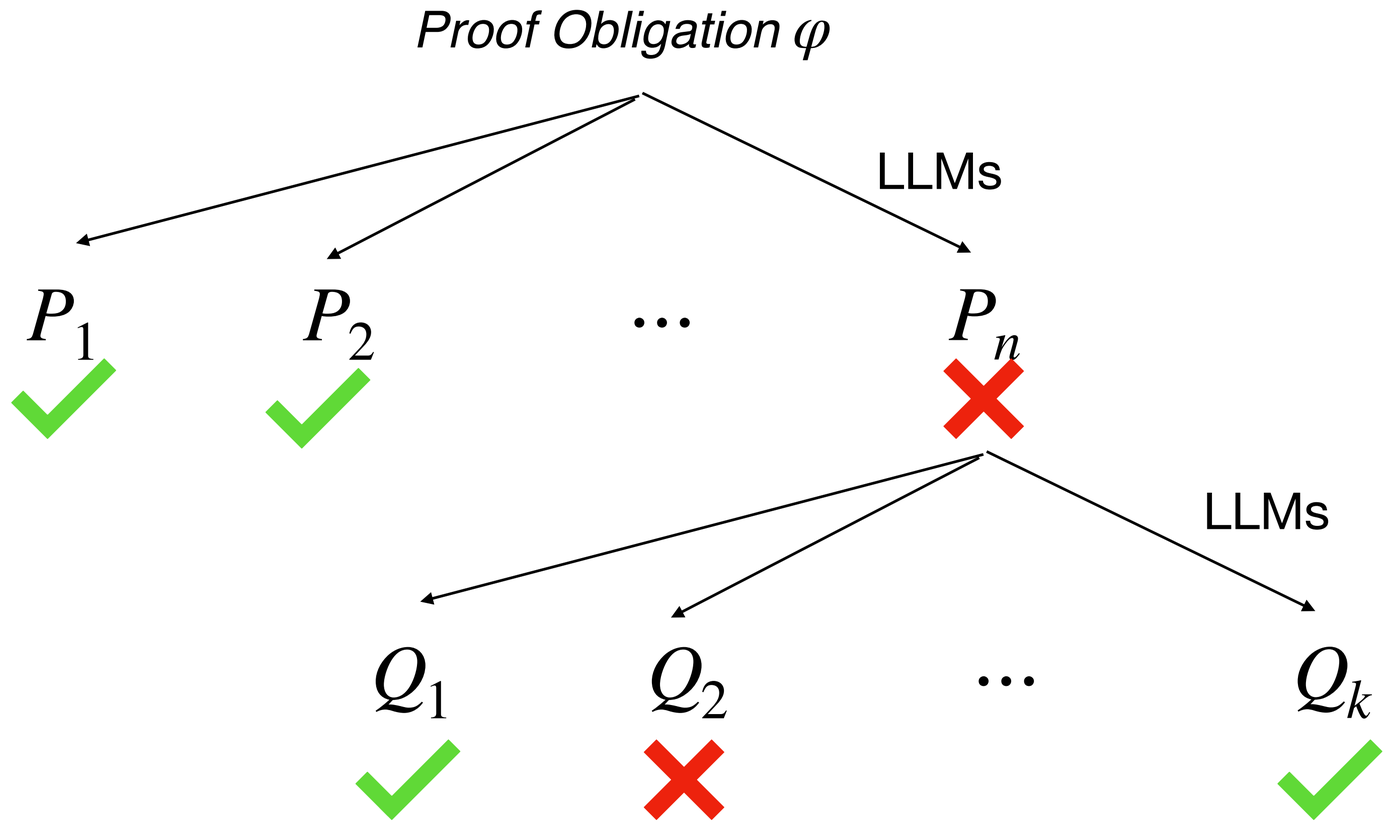
Language Model Guided Obligation Decomposition



SAPG can prove it

SAPG cannot prove it

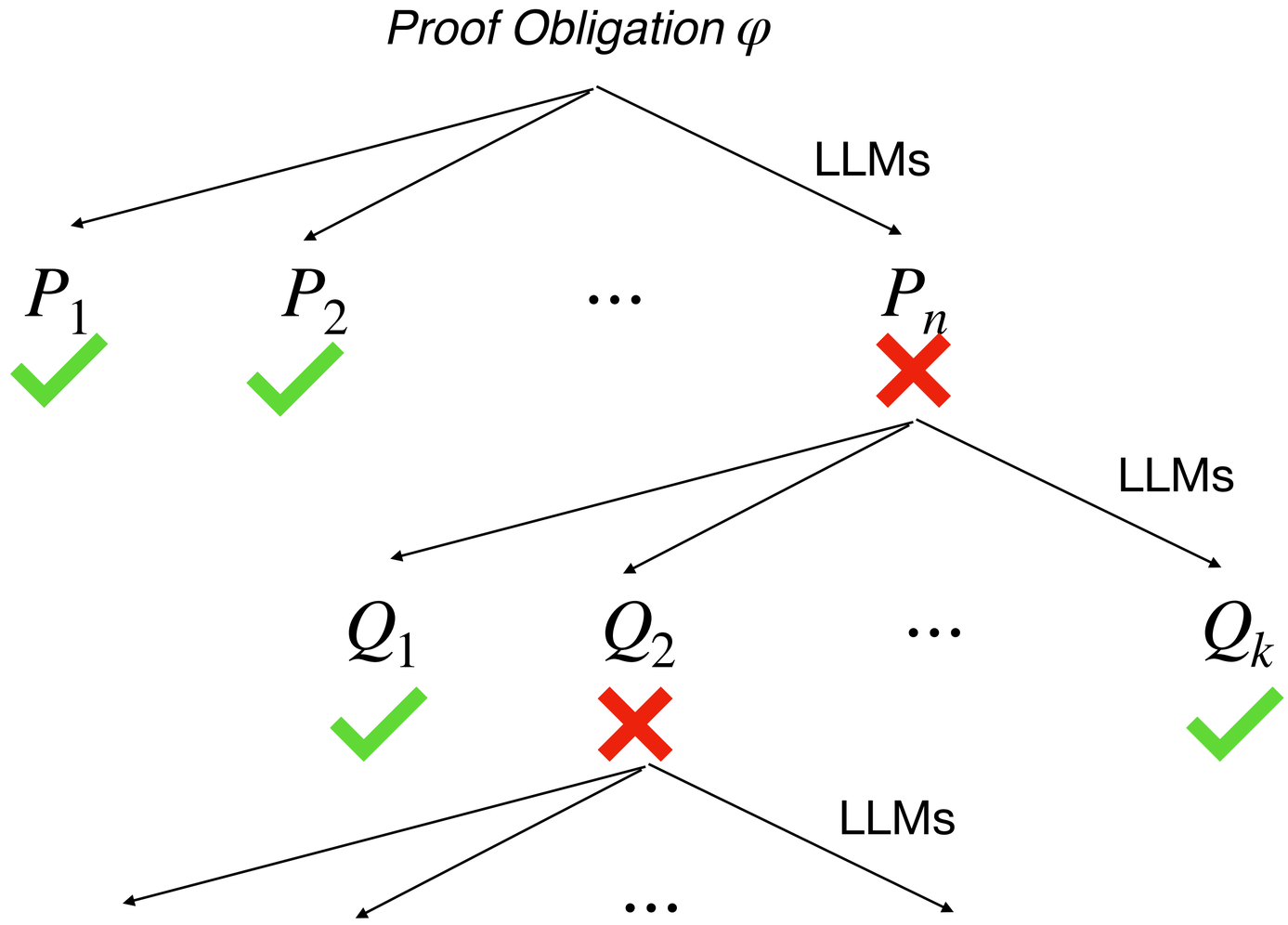
Language Model Guided Obligation Decomposition



SAPG can prove it

SAPG cannot prove it

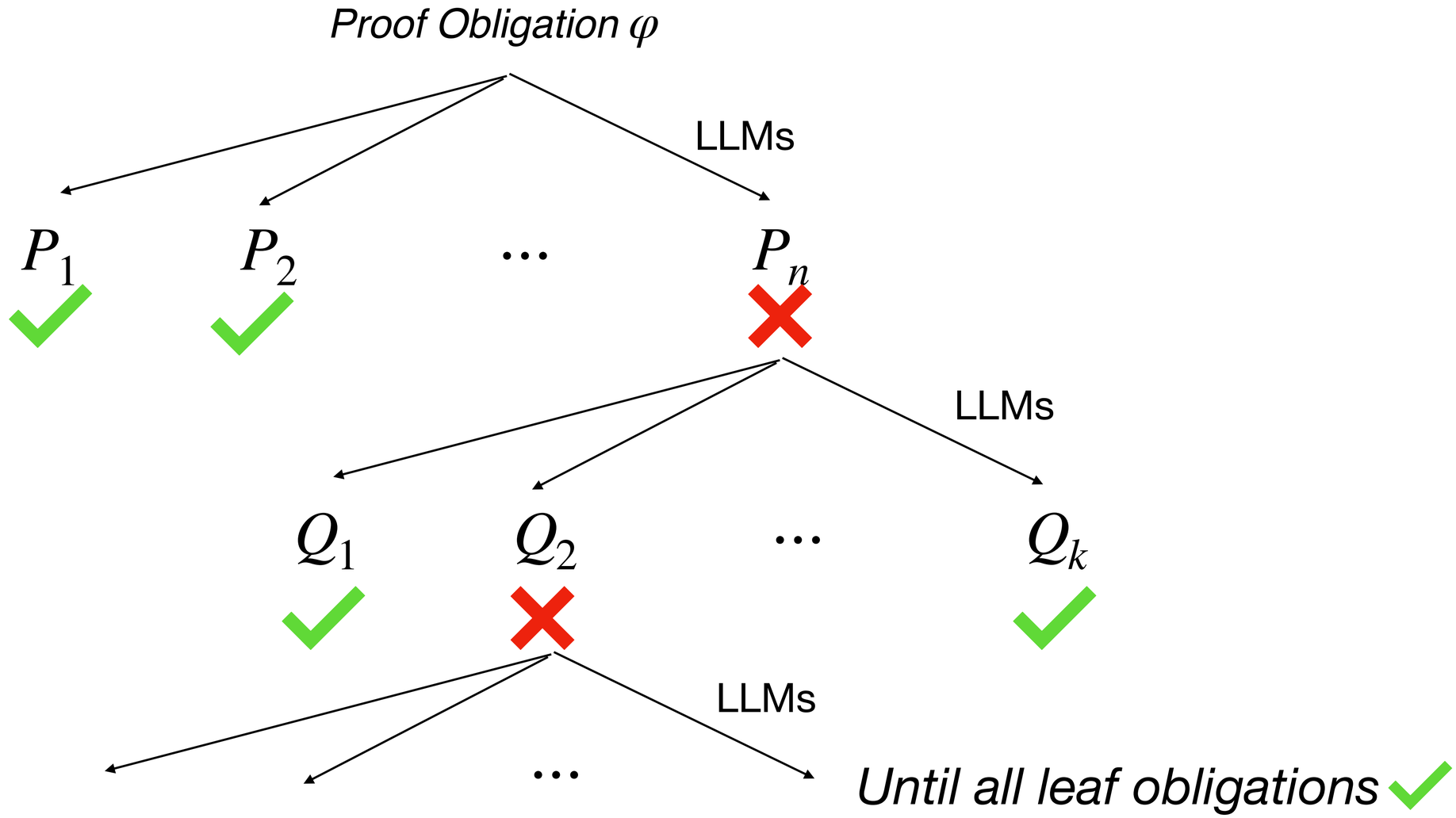
Language Model Guided Obligation Decomposition



SAPG can prove it

SAPG cannot prove it

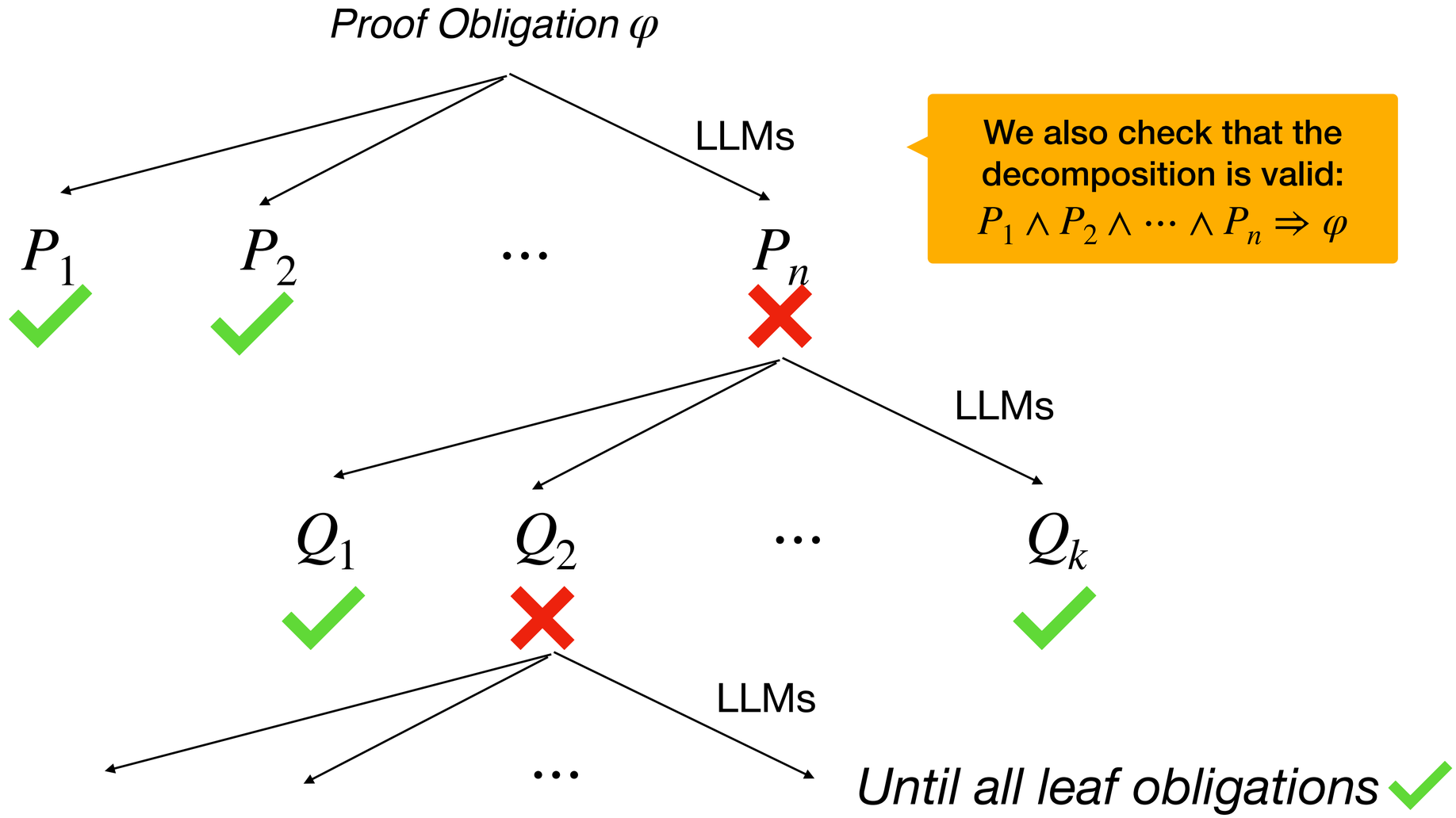
Language Model Guided Obligation Decomposition



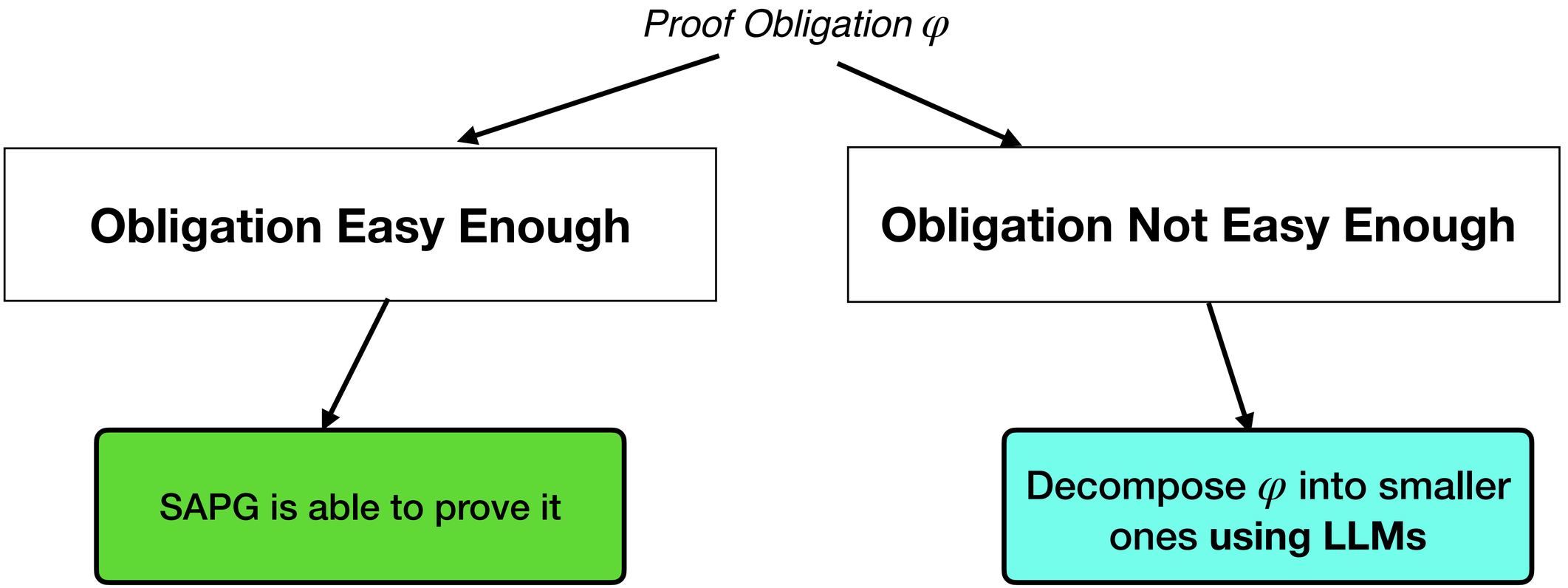
SAPG can prove it

SAPG cannot prove it

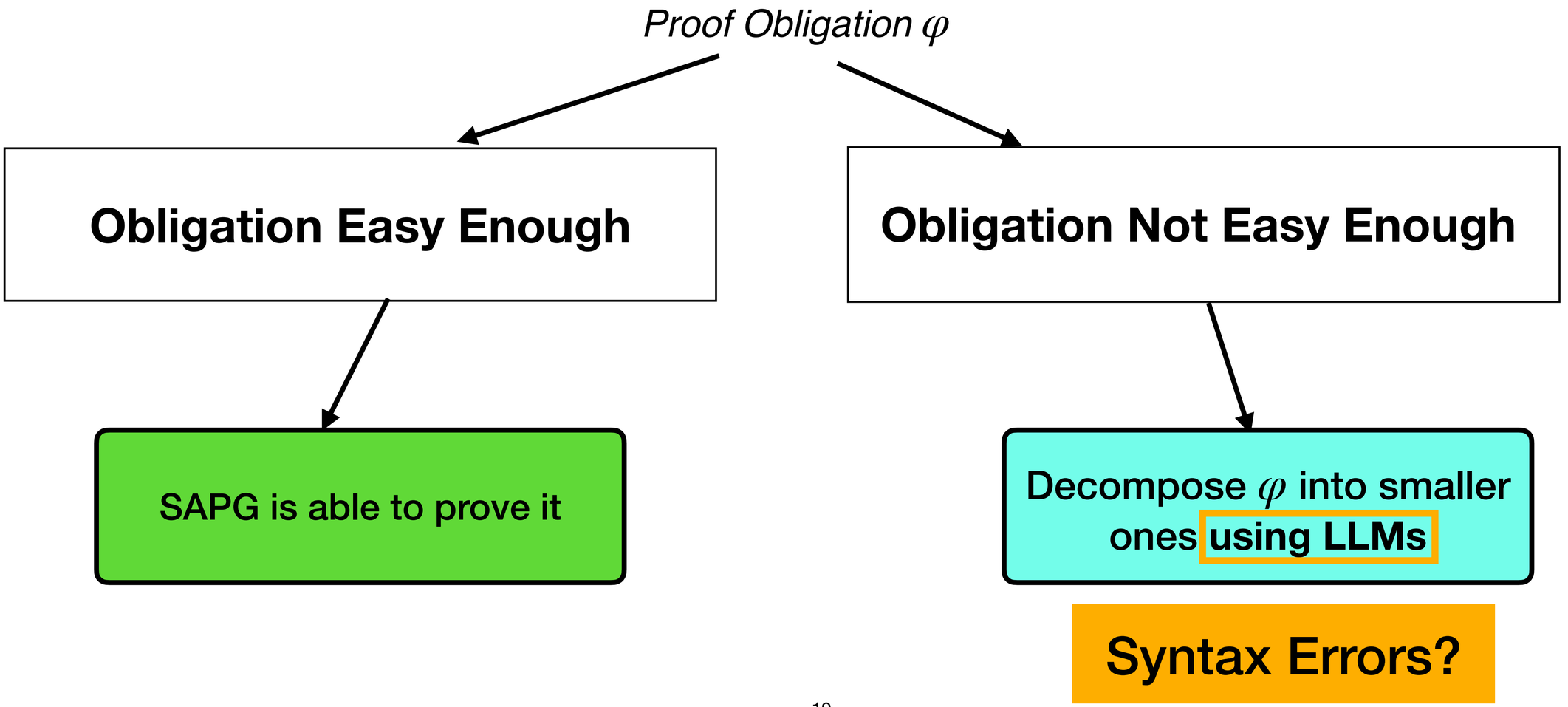
Language Model Guided Obligation Decomposition



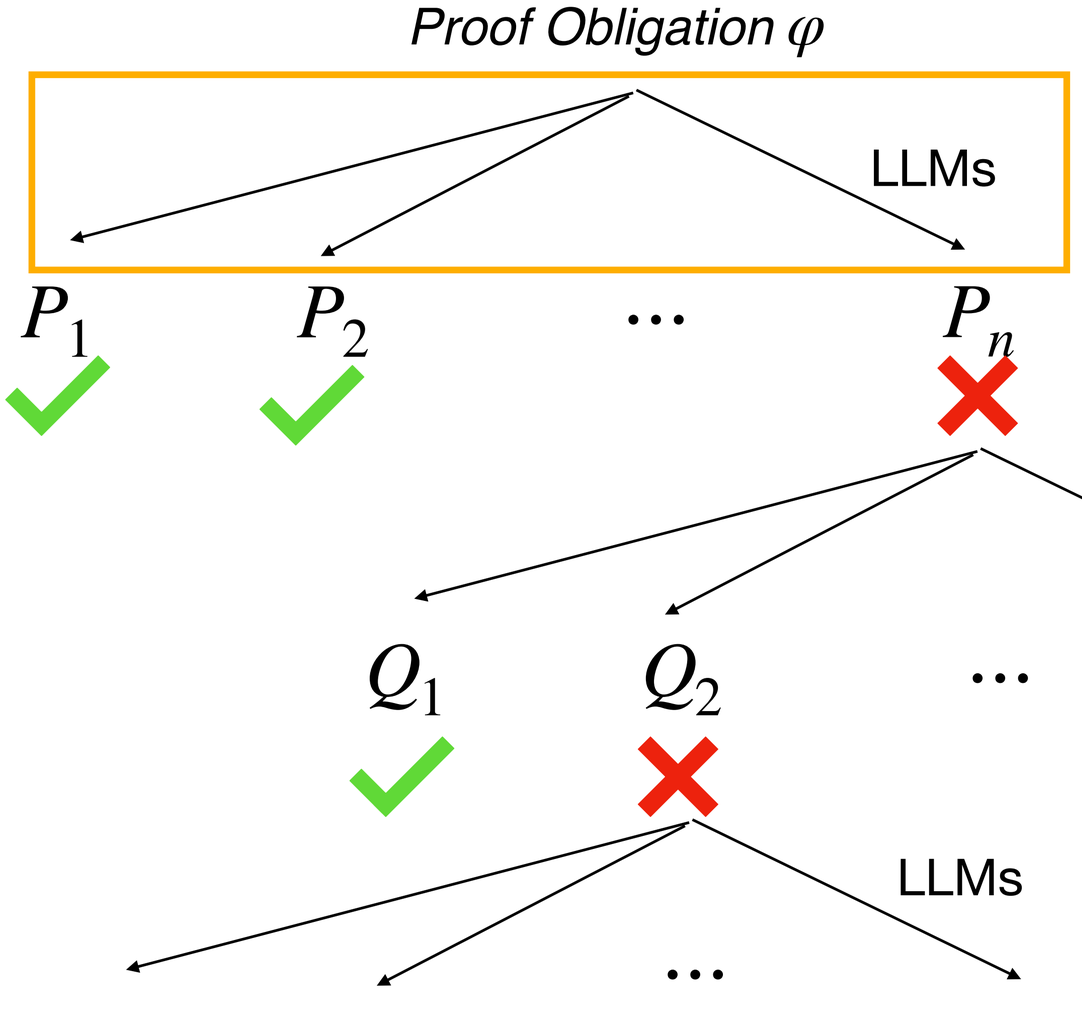
Proving an Obligation



Proving an Obligation



Normalizing Decomposition



We ask the LLMs to generate **normalized** decomposition

✓ SAPG can prove it

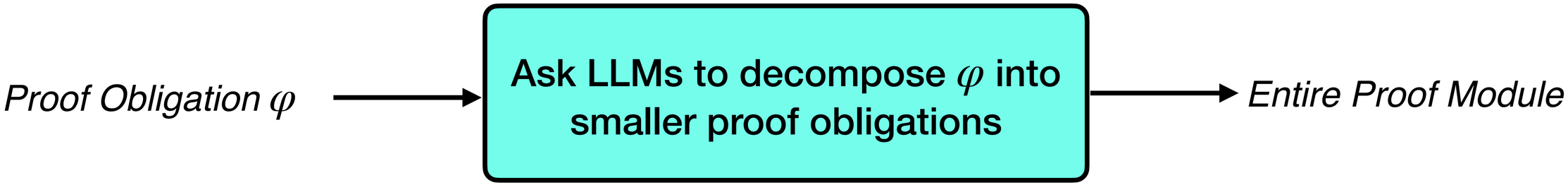
✗ SAPG cannot prove it

Until all obligations ✓

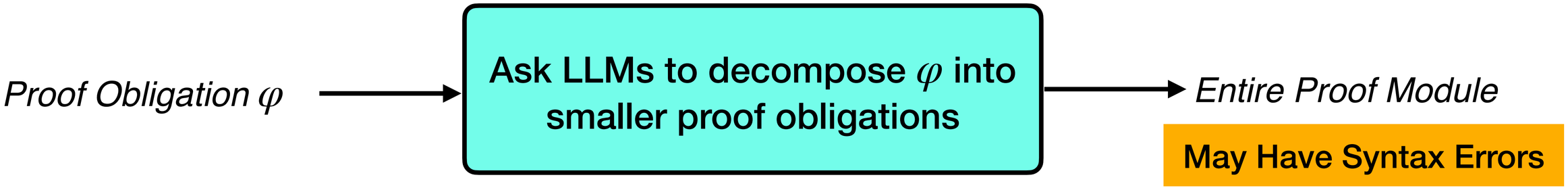
Normalizing Decomposition

Proof Obligation φ

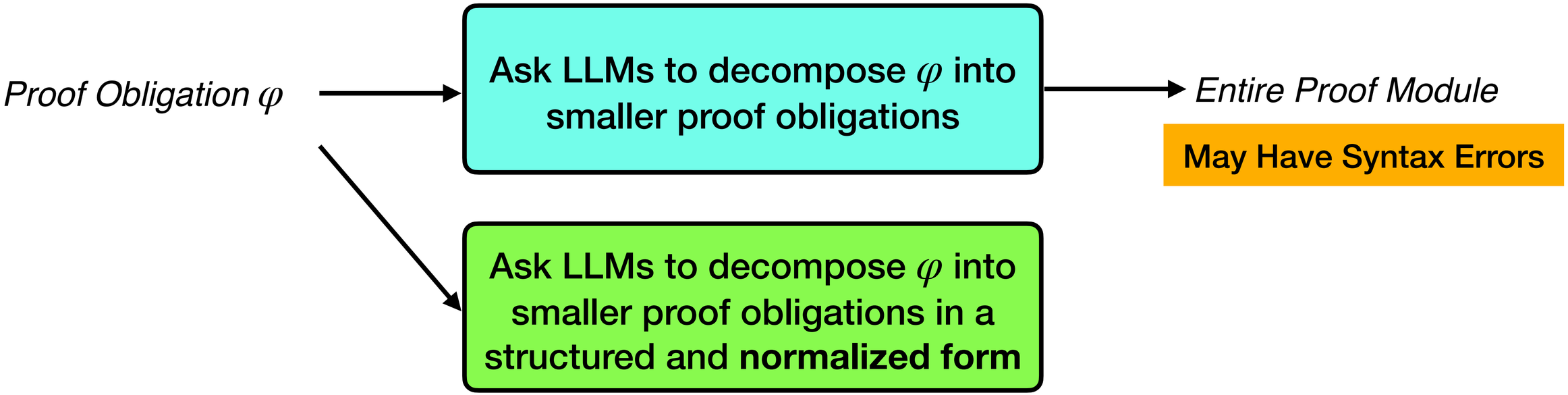
Normalizing Decomposition



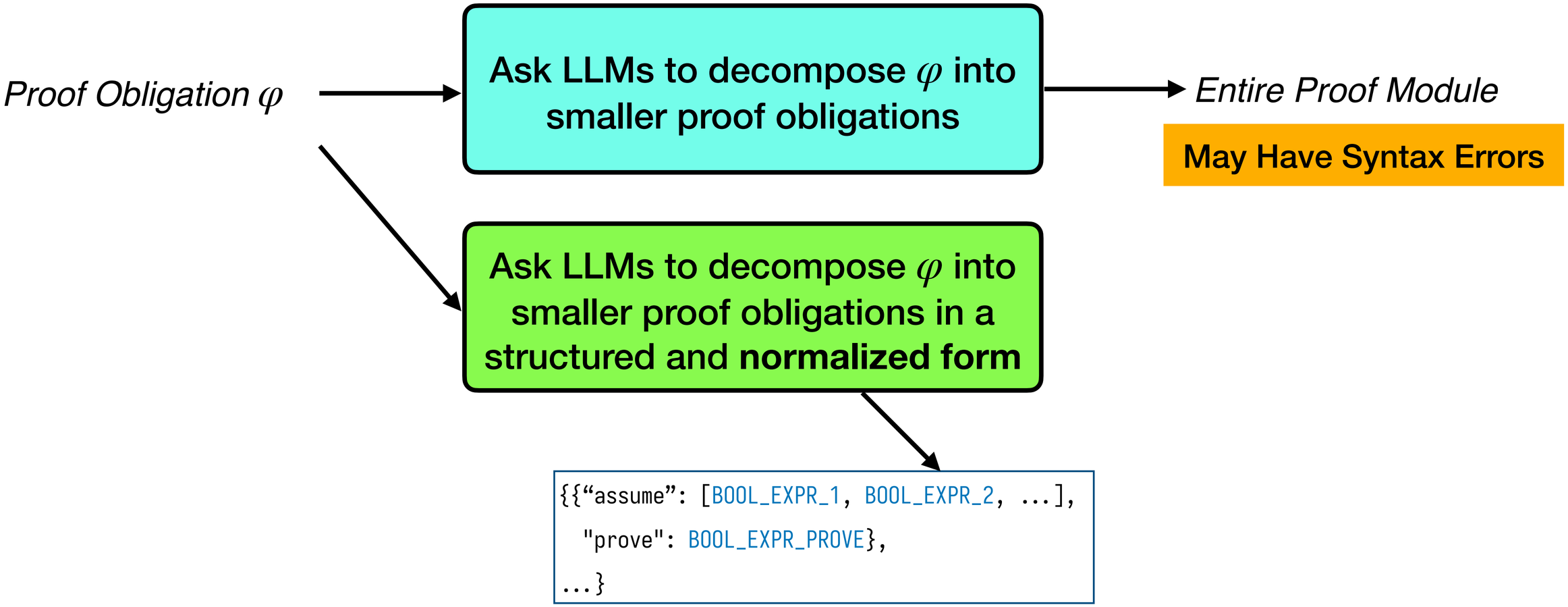
Normalizing Decomposition



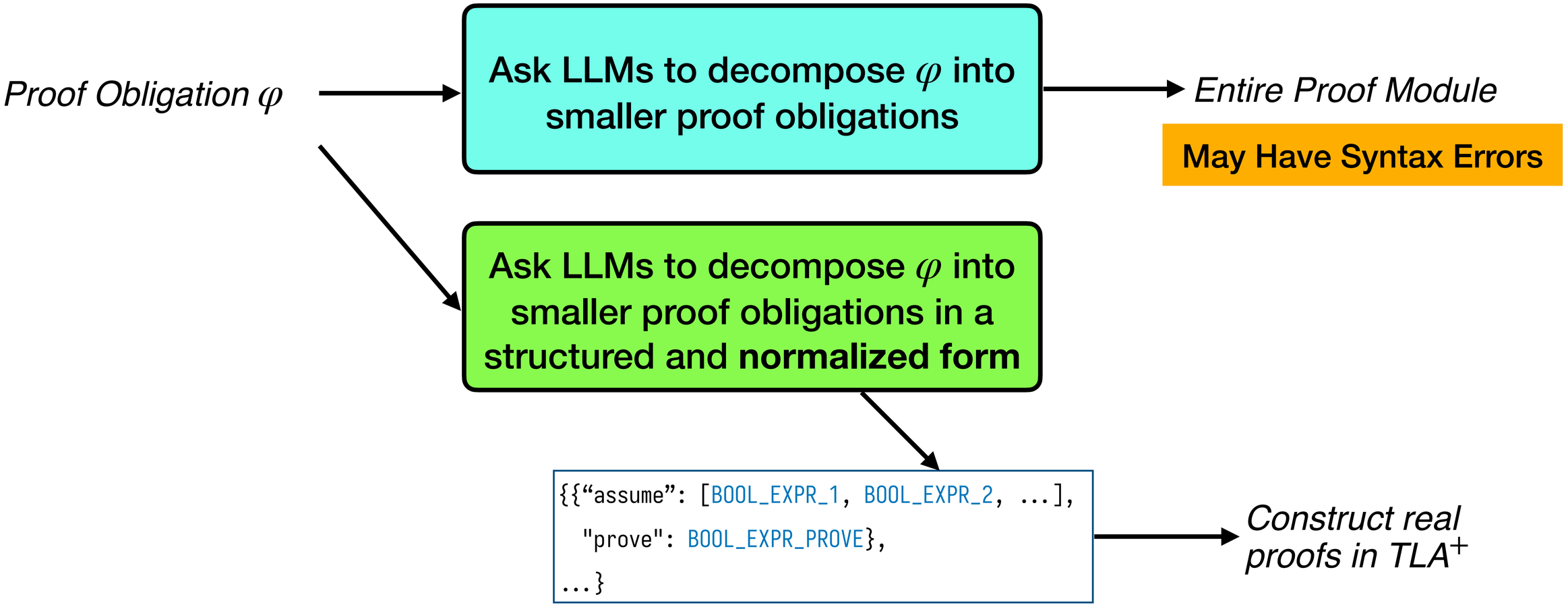
Normalizing Decomposition



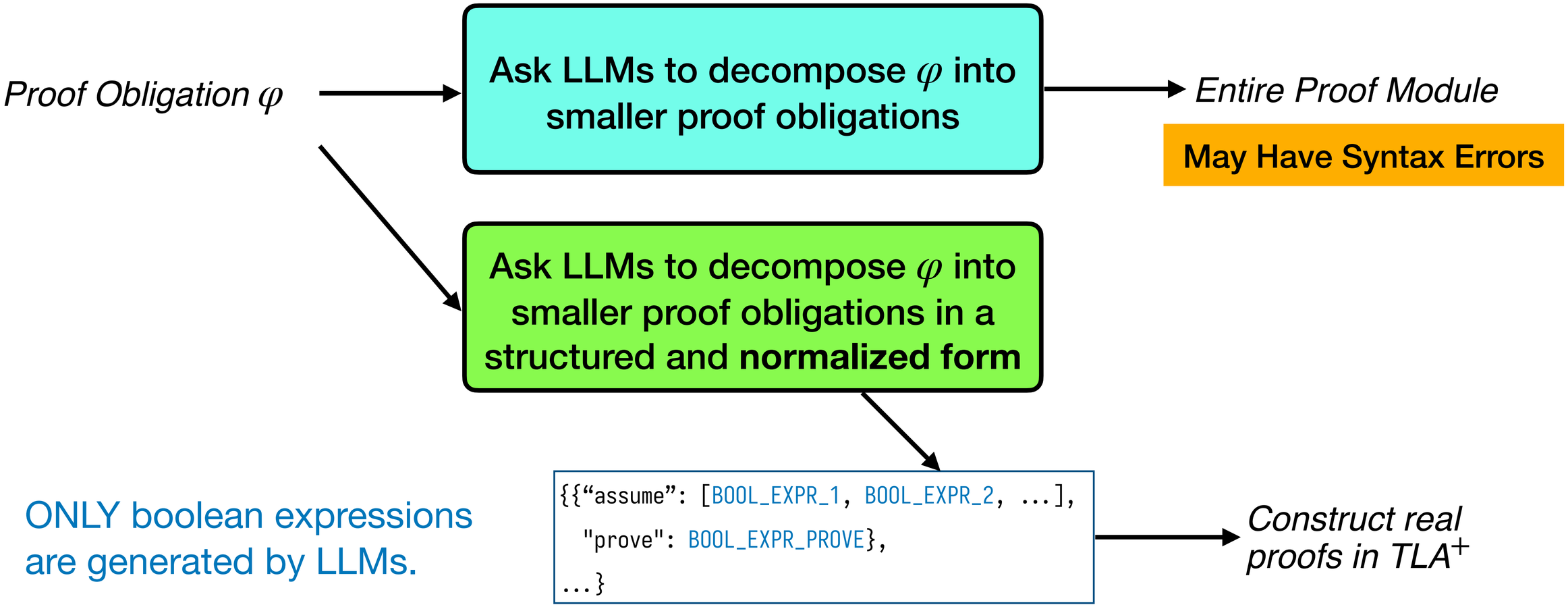
Normalizing Decomposition



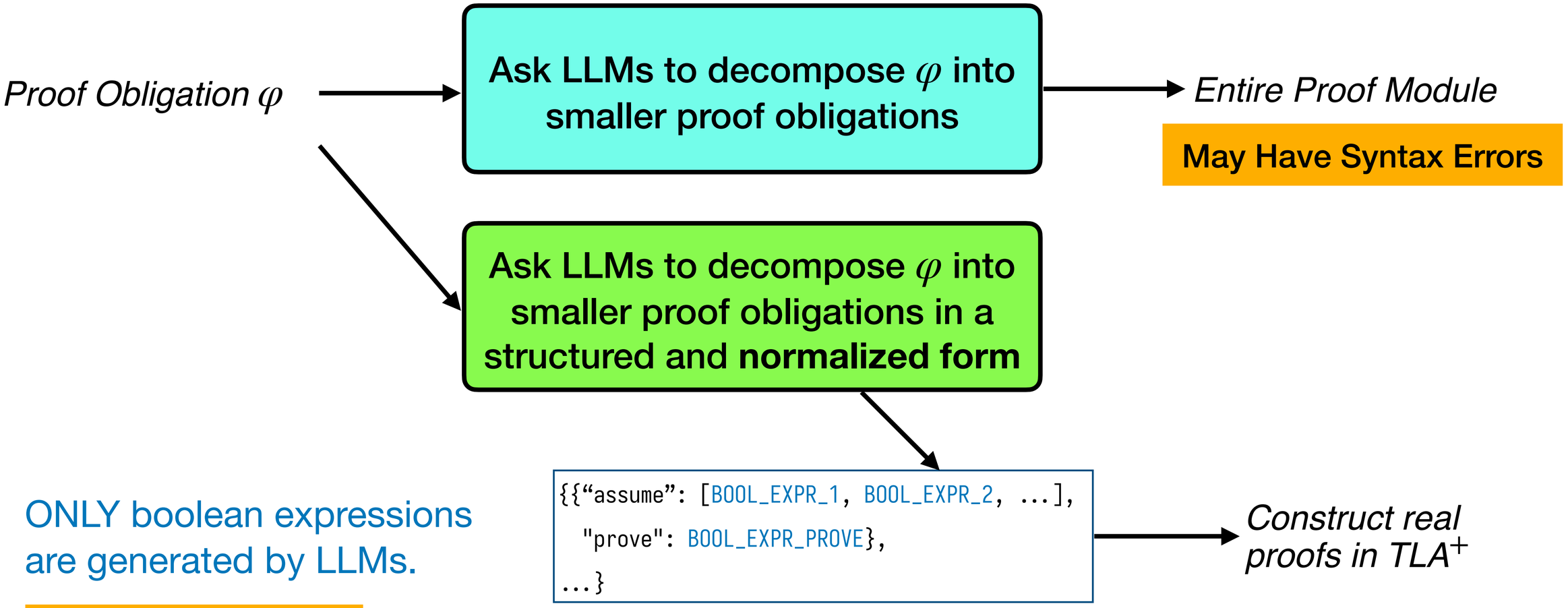
Normalizing Decomposition



Normalizing Decomposition



Normalizing Decomposition



Normalizing Decomposition - Example

Normalizing Decomposition - Example

```
THEOREM T1 == ASSUME NEW x ∈ Nat PROVE
      Even(x + x)
```

Proof Obligation φ

Normalizing Decomposition - Example

```
THEOREM T1 == ASSUME NEW x ∈ Nat PROVE
      Even(x + x)
```

Proof Obligation φ → Ask LLMs to decompose φ into smaller proof obligations in a structured and **normalized form**

Normalizing Decomposition - Example

```
THEOREM T1 == ASSUME NEW x ∈ Nat PROVE
      Even(x + x)
```

Proof Obligation φ

Ask LLMs to decompose φ into smaller proof obligations in a structured and normalized form

```
{{"assume": ["x \in Nat"],
  "prove": "Even(x + x) = Even(x * 2)"},
{"assume": ["x \in Nat"],
  "prove": "Even(x * 2) = ((x * 2) % 2 = 0)"}}
```

Normalizing Decomposition - Example

```
THEOREM T1 == ASSUME NEW x ∈ Nat PROVE
      Even(x + x)
```

Proof Obligation φ

Ask LLMs to decompose φ into smaller proof obligations in a structured and normalized form

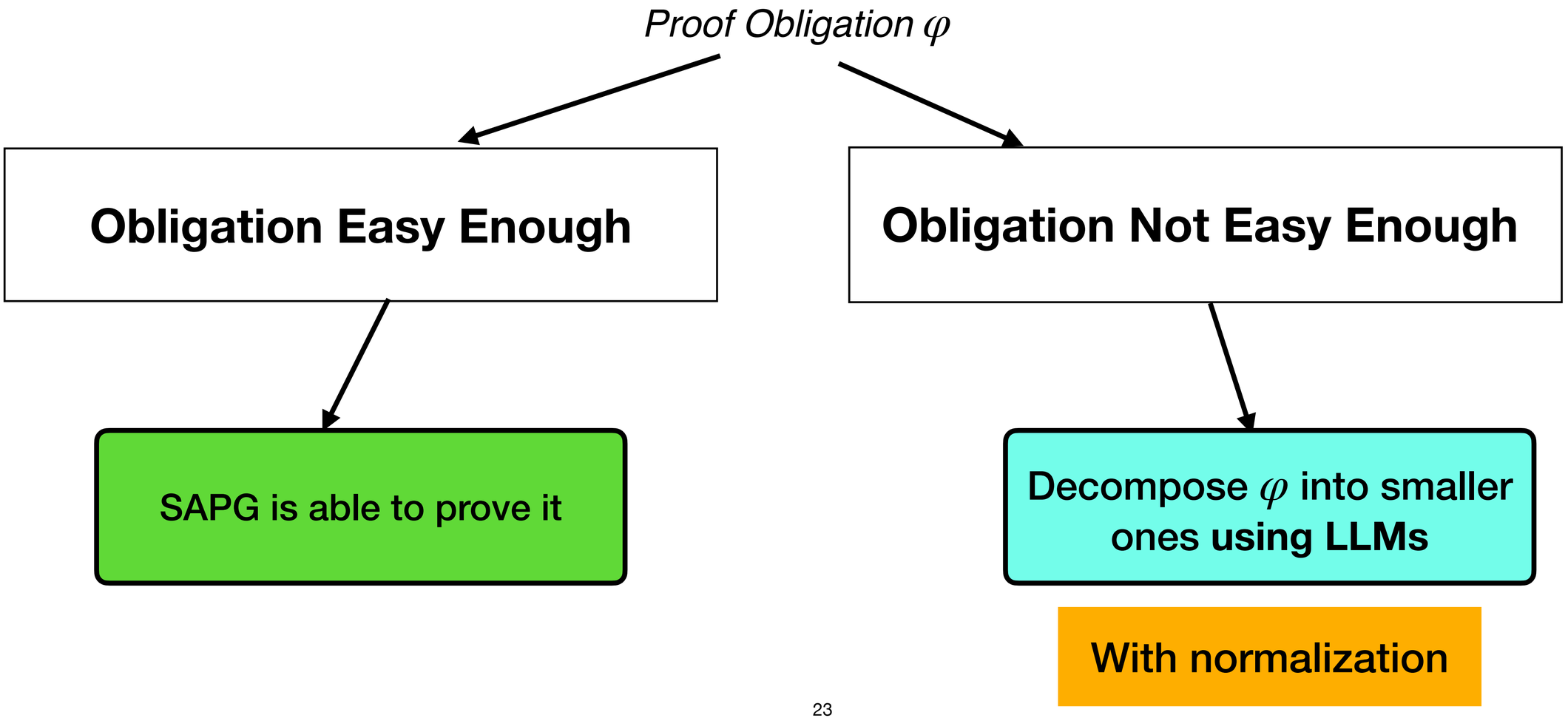
```
{{"assume": ["x \in Nat"],
  "prove": "Even(x + x) = Even(x * 2)"},
{"assume": ["x \in Nat"],
  "prove": "Even(x * 2) = ((x * 2) % 2 = 0)"}}
```

Construct real proofs in TLA$^+$

```
1  -------- MODULE sums_even -------
2  EXTENDS Naturals, TLAPS
3
4  Even(x) == x % 2 = 0
5
6  THEOREM L0 == ASSUME NEW x ∈ Nat PROVE
       Even(x + x) = Even(x * 2)
7  OBVIOUS
8
9  THEOREM L1 == ASSUME NEW x ∈ Nat PROVE
       Even(x * 2) = ((x * 2) % 2 = 0)
10 BY DEF Even
11
12 THEOREM T1 == ASSUME NEW x ∈ Nat PROVE
       Even(x + x)
13 BY L0, L1 DEF Even
14 ======================
```

Language Model Guided Proof Automation

LMGPA



Evaluation

Benchmarks

[1] Kunhao Zheng et al. https://arxiv.org/pdf/2109.00110
[2] Zhangir Azerbayev et al. https://arxiv.org/pdf/2302.12433
[3] William Schultz et al. https://arxiv.org/pdf/2205.06360

Mathematical

93 selected benchmarks manually translated from miniF2F [1] and ProofNet [2]

+

Distributed Protocols

26 inductiveness proofs of candidate inductive invariants of distributed protocols [3]

119 benchmarks in total

Methods to Compare

Methods to Compare

- TLAPS `OBVIOUS` only

Methods to Compare

- TLAPS `OBVIOUS` only
- SAPG

Methods to Compare

- TLAPS `OBVIOUS` only
- SAPG
- Direct LLM Proof Generation (DLPG)

Methods to Compare

- `TLAPS OBVIOUS` only
- SAPG
- Direct LLM Proof Generation (DLPG)
- Language Model Guided Proof Automation (LMGPA)

LLMs Used

LLMs Used

- We selected diverse state-of-the-art open-source and proprietary models:

LLMs Used

- We selected diverse state-of-the-art open-source and proprietary models:
  - Claude 3.7 Sonnet, Deepseek-V3.2-Exp, Gemini 2.0 Flash, Gemini 2.5 Flash, o3-mini-high, and GPT-5

LLMs Used

- We selected diverse state-of-the-art open-source and proprietary models:
  - Claude 3.7 Sonnet, Deepseek-V3.2-Exp, Gemini 2.0 Flash, Gemini 2.5 Flash, o3-mini-high, and GPT-5
- All used without fine-tuning

LLMs Used

- We selected diverse state-of-the-art open-source and proprietary models:
  - Claude 3.7 Sonnet, Deepseek-V3.2-Exp, Gemini 2.0 Flash, Gemini 2.5 Flash, o3-mini-high, and GPT-5
- All used without fine-tuning
- Temperature = 0

Proof Success Rate

Results - Proof Success Rate

	Distributed protocols			Mathematical		
Method	**Proved**	**#Q**	**Time**	**Proved**	**#Q**	**Time**
`TLAPS OBVIOUS`	0.0%	none	1m	44.1%	none	25m
SAPG	38.5%	none	14m	49.5%	none	34m
DLPG[Claude-3.7-Sonnet]	**15.4%**	98	4h 43m	17.2%	321	5h 14m
DLPG[Deepseek-V3.2-Exp]	0.0%	104	11h 27m	**29.0%**	336	39h 51m
DLPG[Gemini-2.0-Flash]	0.0%	104	41m	4.3%	359	39m
DLPG[Gemini-2.5-Flash]	0.0%	104	1h 30m	3.2%	364	3h 18m
DLPG[GPT-5]	3.8%	104	6h 36m	20.7%	307	24h 23m
DLPG[o3-mini-high]	0.0%	104	1h 5m	0.0%	372	8h 30m
LMGPA[Claude-3.7-Sonnet]	42.3%	83	1h 32m	53.8%	231	4h 49m
LMGPA[Deepseek-V3.2-Exp]	**50.0%**	73	9h 14m	**59.1%**	261	33h 17m
LMGPA[Gemini-2.0-Flash]	38.5%	54	39m	54.8%	251	1h 10m
LMGPA[Gemini-2.5-Flash]	46.2%	72	1h 25m	54.8%	224	5h 2m
LMGPA[GPT-5]	42.3%	89	4h 26m	58.1%	245	9h 58m
LMGPA[o3-mini-high]	42.3%	96	1h 44m	57.0%	241	5h 50m

Syntax Errors

Results - Syntax Errors

Model	*DLPG*		*LMGPA*	
	Syn. Valid/#Queries	Percentage	Syn. Valid/#Queries	Percentage
Claude-3.7-Sonnet	88/434	20.3%	206/314	65.6%
Deepseek-V3.2-Exp	84/425	19.8%	287/334	85.9%
Gemini-2.0-Flash	27/463	5.8%	195/305	63.9%
Gemini-2.5-Flash	4/468	0.9%	228/296	77.0%
GPT-5	66/411	16.1%	290/334	86.8%
o3-mini-high	1/476	0.2%	252/337	74.8%

Conclusion

Conclusion

- A recursive LLM-guided decomposition approach for TLA+ proofs
- A prompting strategy that constrains LLMs to generate only normalized decompositions, not full proofs
- A benchmark suite of 119 TLA+ theorems (math + distributed protocols)
- Empirical evaluation showing consistent improvement over baselines

Thank you!