Analyzing Expected Outcomes and (Positive) Almost-Sure Termination of Probabilistic Programs is Hard

Benjamin Kaminski Joost-Pieter Katoen

Software Modeling and Verification Chair

RWTH AACHEN UNIVERSITY

27.2.2015

Motivation

- Probabilistic Programs are like ordinary programs, except:

Motivation

- Probabilistic Programs are like ordinary programs, except:
  - Allow for random choice on how to continue the execution
  - Random choice is done with some specified probability p

Motivation

- Probabilistic Programs are like ordinary programs, except:
  - Allow for random choice on how to continue the execution
  - Random choice is done with some specified probability p
- **Analysis problems** we consider:

Motivation

- Probabilistic Programs are like ordinary programs, except:
  - Allow for random choice on how to continue the execution
  - Random choice is done with some specified probability p
- **Analysis problems** we consider:
  - Determine the value of a variable after program execution

Motivation

- Probabilistic Programs are like ordinary programs, except:
  - Allow for random choice on how to continue the execution
  - Random choice is done with some specified probability p
- **Analysis problems** we consider:
  - Determine the value of a variable after program execution
  - Decide whether the program terminates

Motivation

- Probabilistic Programs are like ordinary programs, except:
  - Allow for random choice on how to continue the execution
  - Random choice is done with some specified probability p
- **Analysis problems** we consider:
  - Determine the value of a variable after program execution
    - Determine expected values
  - Decide whether the program terminates

Motivation

- Probabilistic Programs are like ordinary programs, except:
  - Allow for random choice on how to continue the execution
  - Random choice is done with some specified probability p
- **Analysis problems** we consider:
  - Determine the value of a variable after program execution
    - Determine expected values (expected outcomes)
  - Decide whether the program terminates

Motivation

- Probabilistic Programs are like ordinary programs, except:
  - Allow for random choice on how to continue the execution
  - Random choice is done with some specified probability p
- **Analysis problems** we consider:
  - Determine the value of a variable after program execution
    - Determine expected values (expected outcomes)
  - Decide whether the program terminates
    - Decide almost–sure termination

Motivation

- Probabilistic Programs are like ordinary programs, except:
  - Allow for random choice on how to continue the execution
  - Random choice is done with some specified probability p
- **Analysis problems** we consider:
  - Determine the value of a variable after program execution
    - Determine expected values (expected outcomes)
  - Decide whether the program terminates (in an expected finite number of steps)
    - Decide almost–sure termination

Motivation

- Probabilistic Programs are like ordinary programs, except:
  - Allow for random choice on how to continue the execution
  - Random choice is done with some specified probability p
- **Analysis problems** we consider:
  - Determine the value of a variable after program execution
    - Determine expected values (expected outcomes)
  - Decide whether the program terminates (in an expected finite number of steps)
    - Decide (positive) almost–sure termination

Motivation

- Probabilistic Programs are like ordinary programs, except:
  - Allow for random choice on how to continue the execution
  - Random choice is done with some specified probability p
- **Analysis problems** we consider:
  - Determine the value of a variable after program execution
    - Determine expected values (expected outcomes)
  - Decide whether the program terminates (in an expected finite number of steps) [on all inputs]
    - Decide (positive) almost–sure termination

Motivation

- Probabilistic Programs are like ordinary programs, except:
  - Allow for random choice on how to continue the execution
  - Random choice is done with some specified probability p
- **Analysis problems** we consider:
  - Determine the value of a variable after program execution
    - Determine expected values (expected outcomes)
  - Decide whether the program terminates (in an expected finite number of steps) [on all inputs]
    - Decide [universal] (positive) almost–sure termination

Motivation

- Probabilistic Programs are like ordinary programs, except:
  - Allow for random choice on how to continue the execution
  - Random choice is done with some specified probability p
- **Analysis problems** we consider:
  - Determine the value of a variable after program execution
    - Determine expected values (expected outcomes)
  - Decide whether the program terminates (in an expected finite number of steps) [on all inputs]
    - Decide [universal] (positive) almost–sure termination

How hard is it to solve these analysis problems?

Dissent in the Literature

[Morgan 1996]

"[. . .] probabilistic reasoning for partial correctness [. . .] is not much more complex than standard reasoning."

Dissent in the Literature

[Morgan 1996]

"[. . .] probabilistic reasoning for partial correctness [. . .] is not much more complex than standard reasoning."

[Esparza *et al.* 2012]

"[Ordinary] termination is a purely topological property [. . .], but almost–sure termination is not.
[. . .] proving almost–sure termination requires arithmetic reasoning not offered by termination provers."

The Arithmetical Hierarchy

The Arithmetical Hierarchy

- Class Σ_n^0 is defined as

$$\Sigma_n^0 \;=\; \Big\{ A \;\Big|\; A = \{\vec{x} \mid \exists y_1\, \forall y_2\, \exists y_3 \;\cdots\; \exists/\forall y_n : (\vec{x},\, y_1,\, y_2,\, y_3,\, \ldots,\, y_n) \in R\},\; R \text{ is a decidable relation} \Big\}$$

The Arithmetical Hierarchy

- Class Σ_n^0 is defined as

$$\Sigma_n^0 \;=\; \Big\{ A \;\Big|\; A = \{\vec{x} \mid \exists y_1\, \forall y_2\, \exists y_3 \,\cdots\, \exists/\forall y_n : (\vec{x},\, y_1,\, y_2,\, y_3,\, \ldots,\, y_n) \in R\},\; R \text{ is a decidable relation} \Big\}$$

- Class Π_n^0 is defined as

$$\Pi_n^0 \;=\; \Big\{ A \;\Big|\; A = \{\vec{x} \mid \forall y_1\, \exists y_2\, \forall y_3 \,\cdots\, \exists/\forall y_n : (\vec{x},\, y_1,\, y_2,\, y_3,\, \ldots,\, y_n) \in R\},\; R \text{ is a decidable relation} \Big\}$$

The Arithmetical Hierarchy

- Class Σ_n^0 is defined as

$$\Sigma_n^0 \;=\; \Big\{ A \;\Big|\; A = \{\vec{x} \mid \exists y_1\, \forall y_2\, \exists y_3 \,\cdots\, \exists/\forall y_n \colon (\vec{x}, y_1, y_2, y_3, \ldots, y_n) \in R\}, \; R \text{ is a decidable relation} \Big\}$$

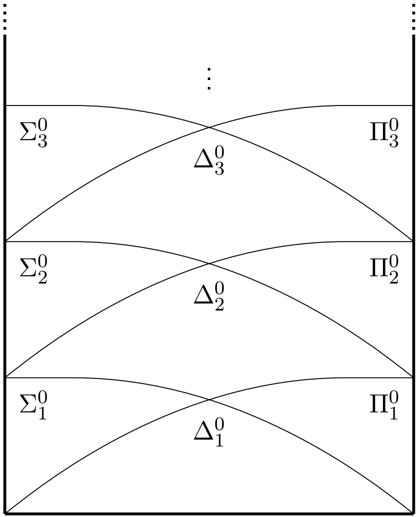
- Class Π_n^0 is defined as

$$\Pi_n^0 \;=\; \Big\{ A \;\Big|\; A = \{\vec{x} \mid \forall y_1\, \exists y_2\, \forall y_3 \,\cdots\, \exists/\forall y_n \colon (\vec{x}, y_1, y_2, y_3, \ldots, y_n) \in R\}, \; R \text{ is a decidable relation} \Big\}$$

- Class Δ_n^0 is defined as $\Delta_n^0 = \Sigma_n^0 \cap \Pi_n^0$

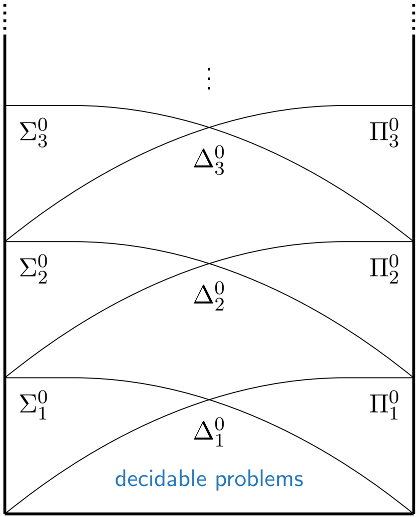
The Arithmetical Hierarchy — The Bigger Picture

The following inclusion diagram holds (all inclusions are strict):



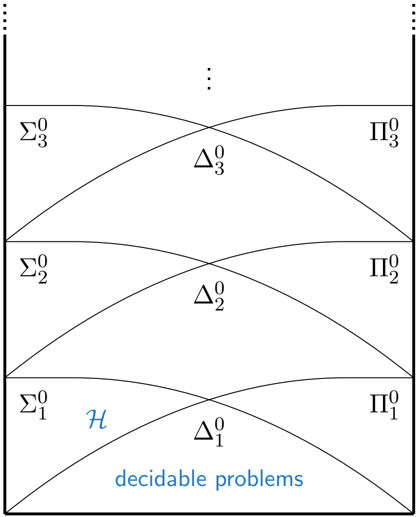
The Arithmetical Hierarchy — The Bigger Picture

The following inclusion diagram holds (all inclusions are strict):



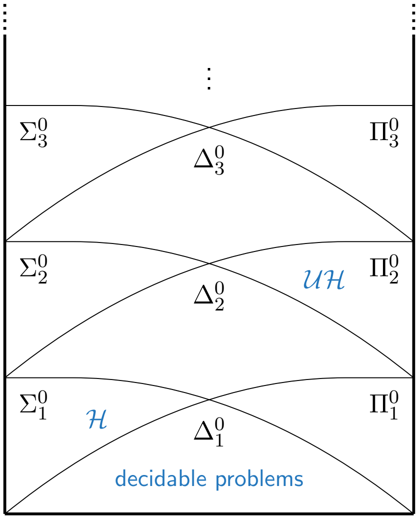
The Arithmetical Hierarchy — The Bigger Picture

The following inclusion diagram holds (all inclusions are strict):



The Arithmetical Hierarchy — The Bigger Picture

The following inclusion diagram holds (all inclusions are strict):



Some Notation

Some Notation

- The expected outcome of variable v after executing P:

$$\mathsf{E}_P(v)$$

Some Notation

- The expected outcome of variable v after executing P:

$$\mathsf{E}_P(v)$$

- The probability that P terminates on input η:

$$\mathsf{Pr}_{P,\eta}(\downarrow)$$

Some Notation

- The expected outcome of variable v after executing P:

$$\mathsf{E}_P(v)$$

- The probability that P terminates on input η:

$$\mathsf{Pr}_{P,\eta}(\downarrow)$$

- The expected number of steps until P terminates on input η:

$$\mathsf{E}_{P,\eta}(\downarrow)$$

Decision Problems We Analyzed

Decision Problems We Analyzed

Lower and Upper Bounds, and Exact Expected Outcomes

$$
\begin{aligned}
(P,\, v,\, q) \in \mathcal{LEXP} \quad &:\Longleftrightarrow \quad q < \mathsf{E}_P(v) \\
(P,\, v,\, q) \in \mathcal{UEXP} \quad &:\Longleftrightarrow \quad q > \mathsf{E}_P(v) \\
(P,\, v,\, q) \in \mathcal{EXP} \quad &:\Longleftrightarrow \quad q = \mathsf{E}_P(v)
\end{aligned}
$$

Decision Problems We Analyzed

Lower and Upper Bounds, and Exact Expected Outcomes

$$(P,\, v,\, q) \in \mathcal{LEXP} \quad :\Longleftrightarrow \quad q < \mathsf{E}_P(v)$$
$$(P,\, v,\, q) \in \mathcal{UEXP} \quad :\Longleftrightarrow \quad q > \mathsf{E}_P(v)$$
$$(P,\, v,\, q) \in \mathcal{EXP} \quad :\Longleftrightarrow \quad q = \mathsf{E}_P(v)$$

Almost–Sure Termination $\mathcal{AST}$

$$(P,\, \eta) \in \mathcal{AST} \;:\Longleftrightarrow\; \mathsf{Pr}_{P,\eta}(\downarrow) = 1$$

Variations of $\mathcal{AST}$

Variations of $\mathcal{AST}$

Positive Almost–Sure Termination $\mathcal{PAST}$

$$(P, \eta) \in \mathcal{PAST} \;:\Longleftrightarrow\; \mathsf{E}_{P,\eta}(\downarrow) < \infty$$

Variations of $\mathcal{AST}$

Positive Almost–Sure Termination $\mathcal{PAST}$

$$(P, \eta) \in \mathcal{PAST} \;:\Longleftrightarrow\; \mathsf{E}_{P,\eta}(\downarrow) < \infty$$

Notice $\mathcal{PAST} \subsetneq \mathcal{AST}$.

Variations of $\mathcal{AST}$

Positive Almost–Sure Termination $\mathcal{PAST}$

$$(P,\, \eta) \in \mathcal{PAST} \;:\Longleftrightarrow\; \mathsf{E}_{P,\eta}(\downarrow) < \infty$$

Notice $\mathcal{PAST} \subsetneq \mathcal{AST}$.

Universal Versions of $\mathcal{AST}$ and $\mathcal{PAST}$

$$P \in \mathcal{UAST} \;:\Longleftrightarrow \forall \eta\colon (P,\, \eta) \in \mathcal{AST}$$
$$P \in \mathcal{UPAST} \;:\Longleftrightarrow \forall \eta\colon (P,\, \eta) \in \mathcal{PAST}$$

A (very) Simple Example Program

Consider the program P_{geo}:

x := 0;
$\{$continue := 0$\}$ [0.5] $\{$continue := 1$\}$;
while (continue $\neq$ 0)$\{$
    x := x + 1;
    $\{$continue := 0$\}$ [0.5] $\{$continue := 1$\}$
$\}$

A (very) Simple Example Program

Consider the program P_{geo}:

```
x := 0;
{continue := 0} [0.5] {continue := 1};
while (continue ≠ 0){
    x := x + 1;
    {continue := 0} [0.5] {continue := 1}
}
```

- $\mathsf{E}_{P_{geo}}(x) = 2$

A (very) Simple Example Program

Consider the program P_{geo}:

```
x := 0;
{continue := 0} [0.5] {continue := 1};
while (continue ≠ 0){
    x := x + 1;
    {continue := 0} [0.5] {continue := 1}
}
```

- $\mathsf{E}_{P_{geo}}(x) = 2$
- $\mathsf{E}_{P_{geo}}(\texttt{continue}) = 0$

A (very) Simple Example Program

Consider the program P_{geo}:

```
x := 0;
{continue := 0} [0.5] {continue := 1};
while (continue ≠ 0){
    x := x + 1;
    {continue := 0} [0.5] {continue := 1}
}
```

- $\mathsf{E}_{P_{geo}}(x) = 2$
- $\mathsf{E}_{P_{geo}}(\texttt{continue}) = 0$
- P_{geo} terminates almost–surely on all inputs

A (very) Simple Example Program

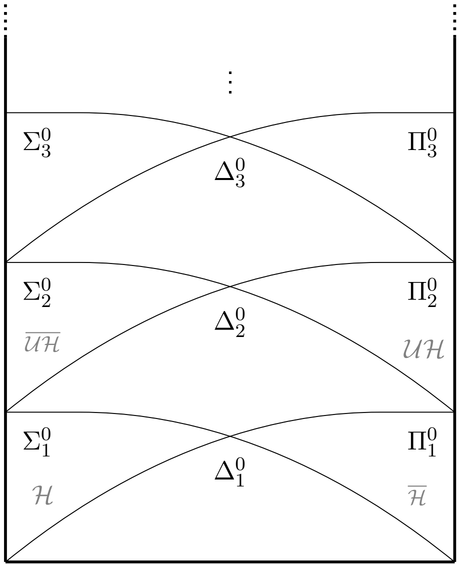
Consider the program P_{geo}:

```
x := 0;
{continue := 0} [0.5] {continue := 1};
while (continue ≠ 0){
    x := x + 1;
    {continue := 0} [0.5] {continue := 1}
}
```

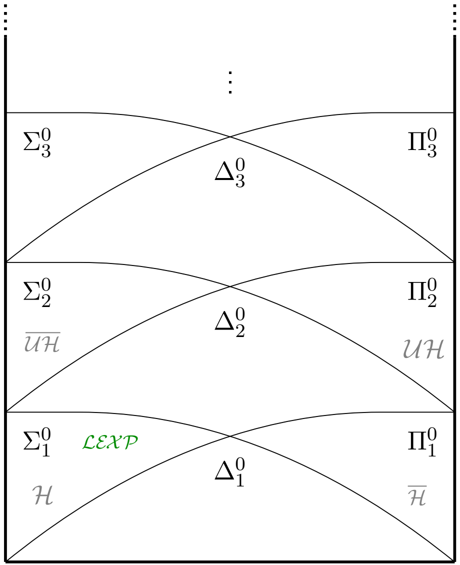
- $\mathsf{E}_{P_{geo}}(x) = 2$
- $\mathsf{E}_{P_{geo}}(\texttt{continue}) = 0$
- P_{geo} terminates almost–surely on all inputs
- Expected runtime of P_{geo} is $\mathcal{O}\left(\mathsf{E}_{P_{geo}}(x)\right)$ on all inputs

Summary of Results

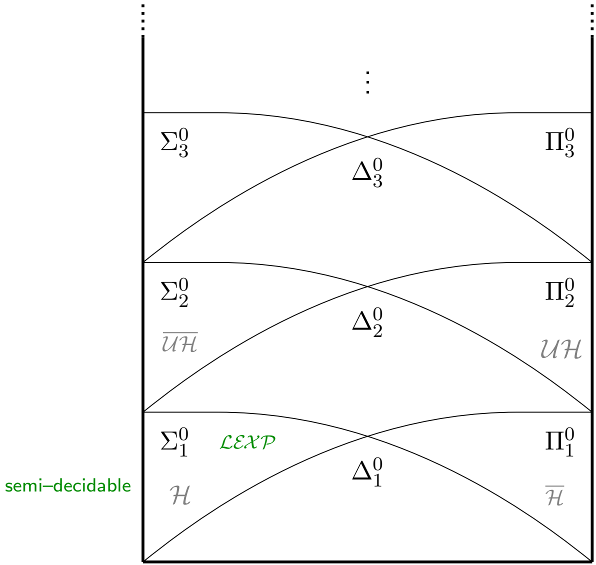
Summary of Results



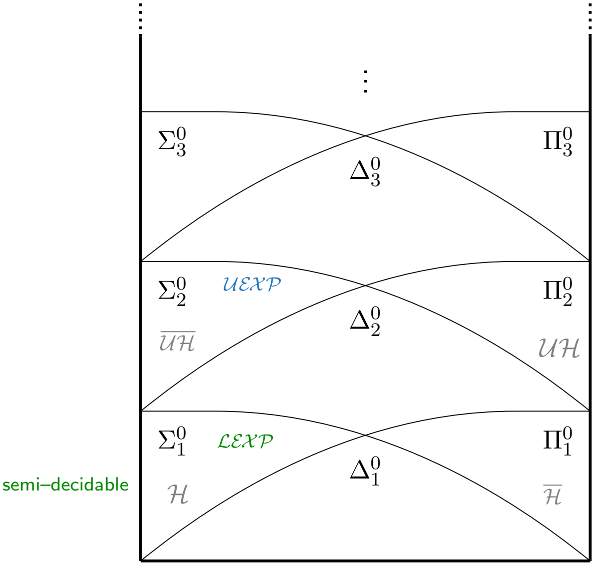
Summary of Results



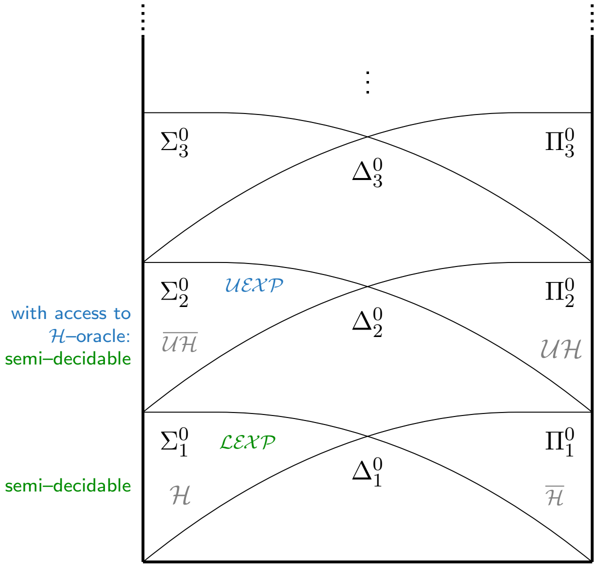
Summary of Results



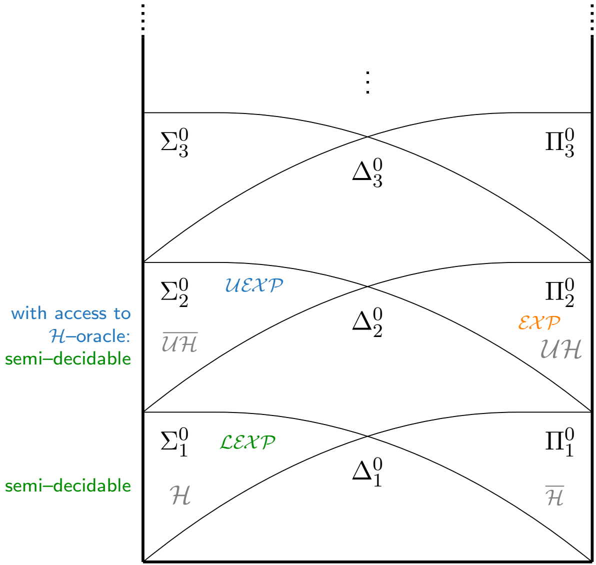
Summary of Results



Summary of Results



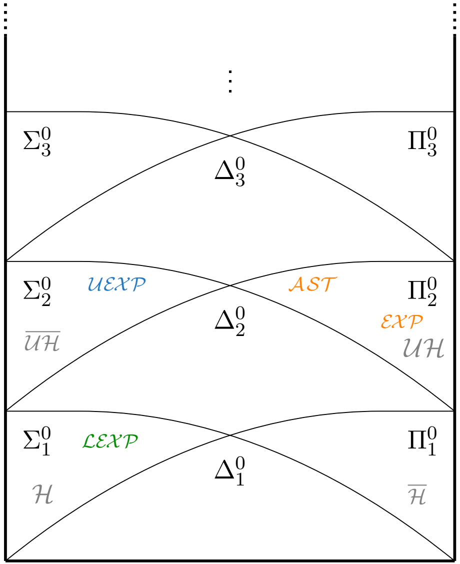
Summary of Results



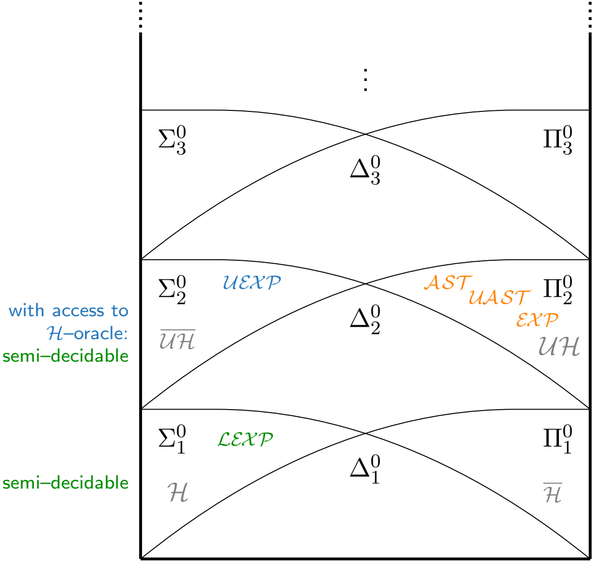
Summary of Results

with access to $\mathcal{H}$–oracle:
semi–decidable

semi–decidable



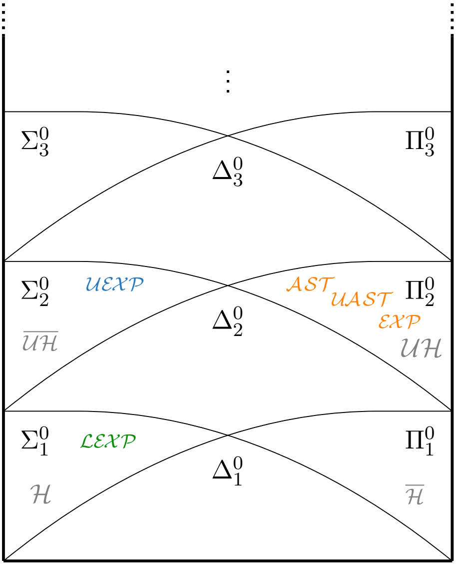
Summary of Results



Summary of Results

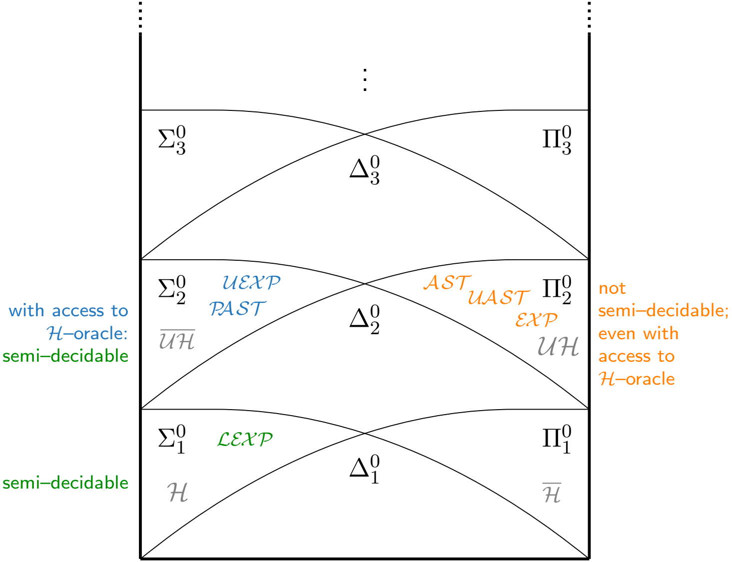
with access to $\mathcal{H}$–oracle: semi–decidable

semi–decidable

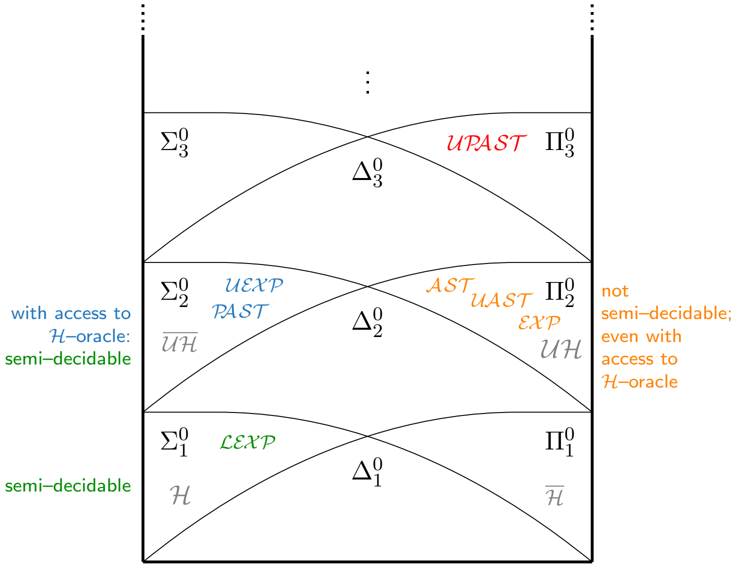


not semi–decidable; even with access to $\mathcal{H}$–oracle

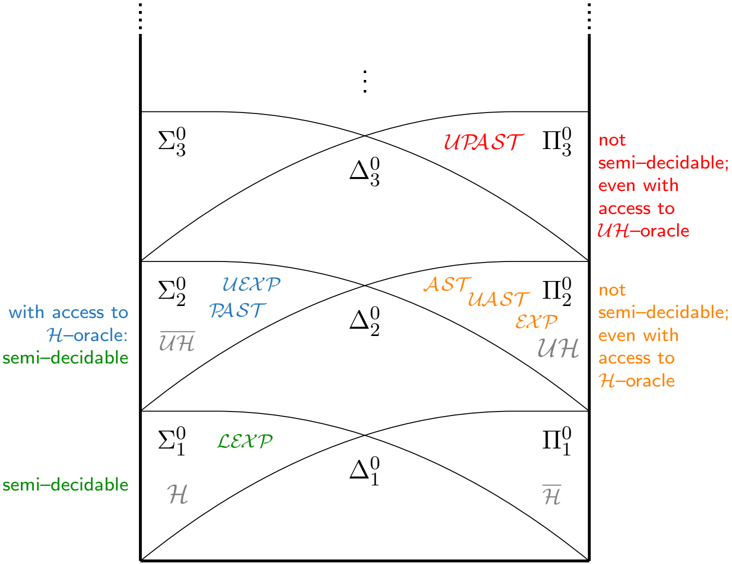
Summary of Results



Summary of Results



Summary of Results



Summary of Results

Thank you for your kind attention :-)