A Progress Bar for Static Analyzers

Woosuk Lee, Hakjoo Oh, and Kwangkeun Yi

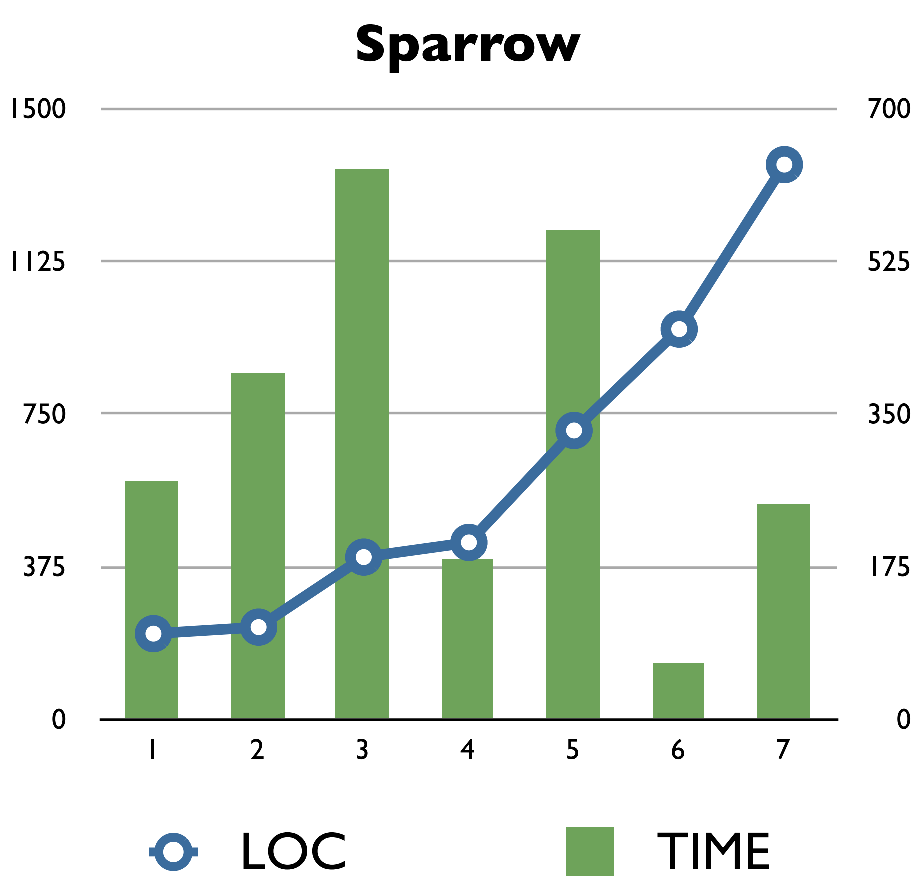
Seoul National University

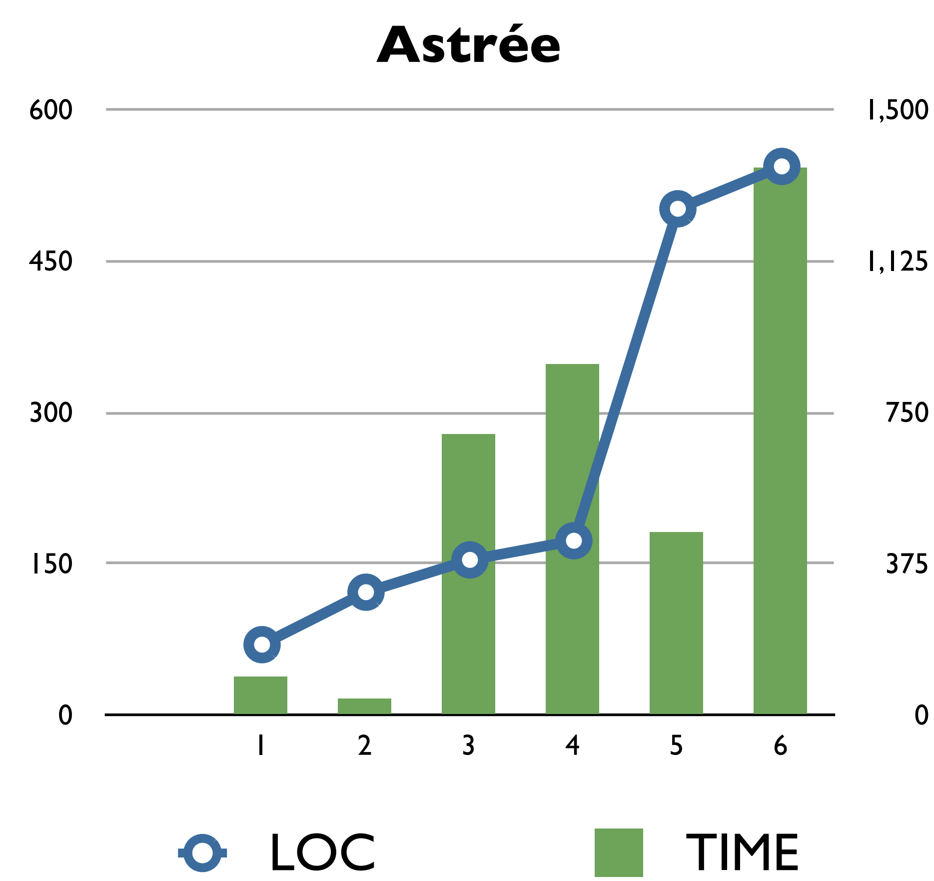
Motivation

- Static analysis of large, complex SW takes a long time.
  - Sparrow - 10 hrs for 400KLOC
  - Astrée - 32 hrs for 780KLOC
  - CGS - 20 hrs for 550KLOC
- One useful UI is missing : a progress indicator

Hard to Estimate

- Analysis time is NOT proportional to program size.





Our Solution

- pre-analysis + machine learning
- generally applicable to abstract interpreters
- shows its applicability for numerical analyses and a pointer analysis on a suit of real C benchmarks

Demo

- Buffer overflow analysis on GNU tar-1.13

Our Goal

- Abstract domain $\mathbb{D}$, semantic function $F : \mathbb{D} \to \mathbb{D}$
- Analysis computes (until stabilized)

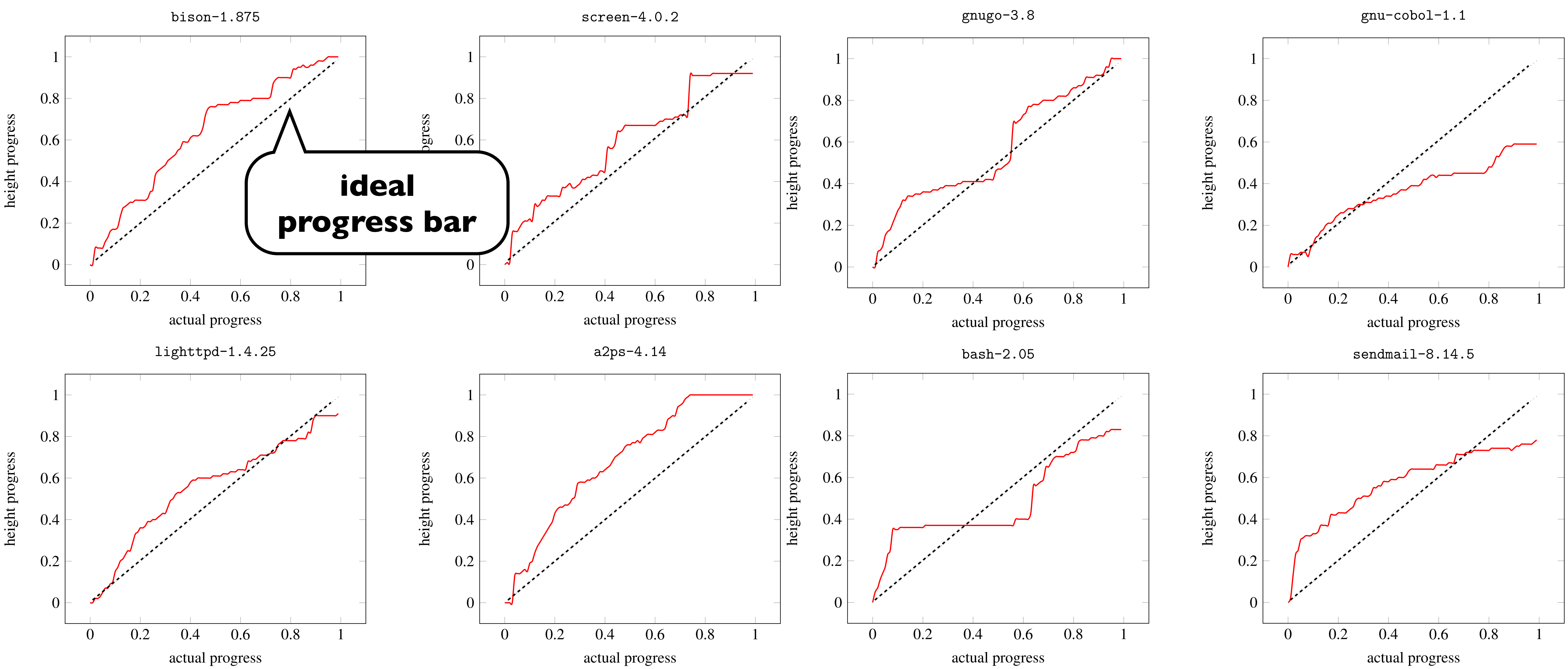
$$\bigsqcup_{i \in \mathbb{N}} F^i(\bot) = F^0(\bot) \sqcup F^1(\bot) \sqcup F^2(\bot) \sqcup \cdots$$

- Ideal progress bar :

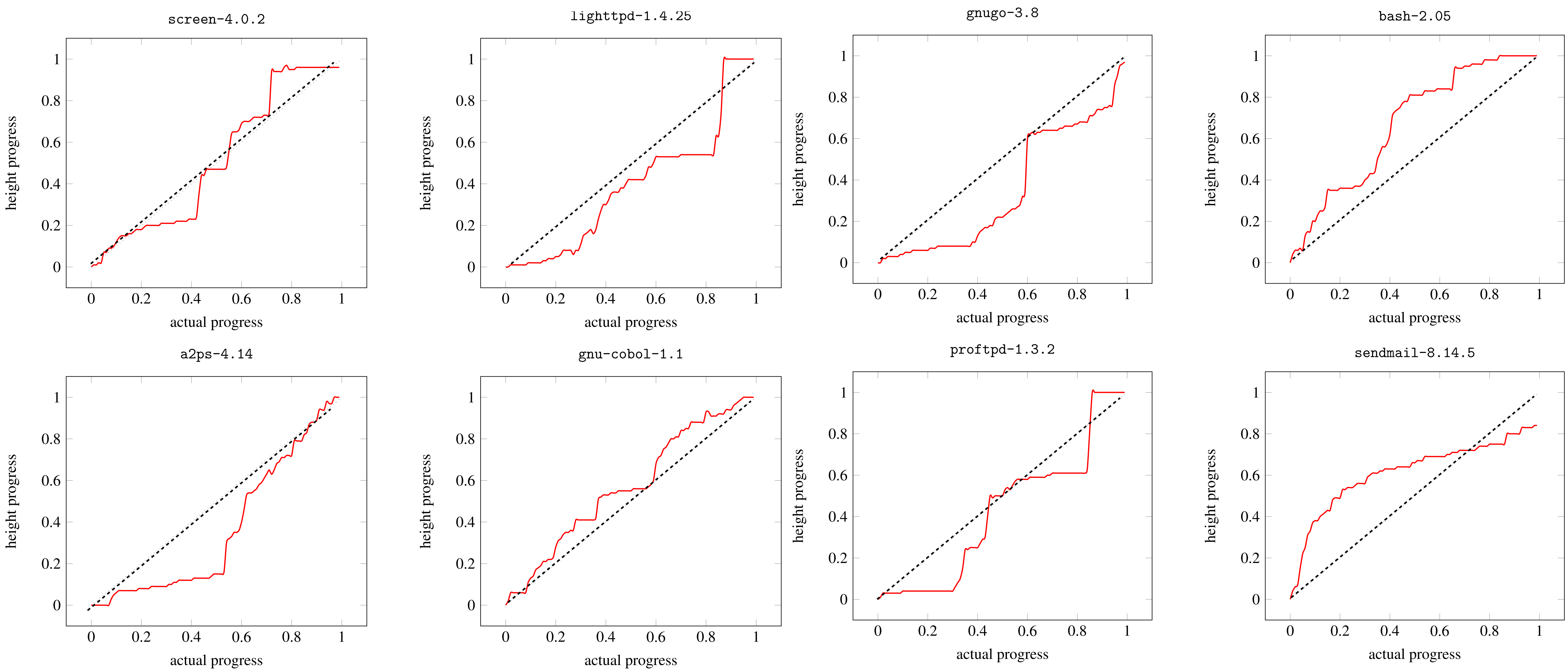
$$\frac{\#\ \text{iterations so far}}{\#\ \text{total iterations}}$$

Result (Interval)

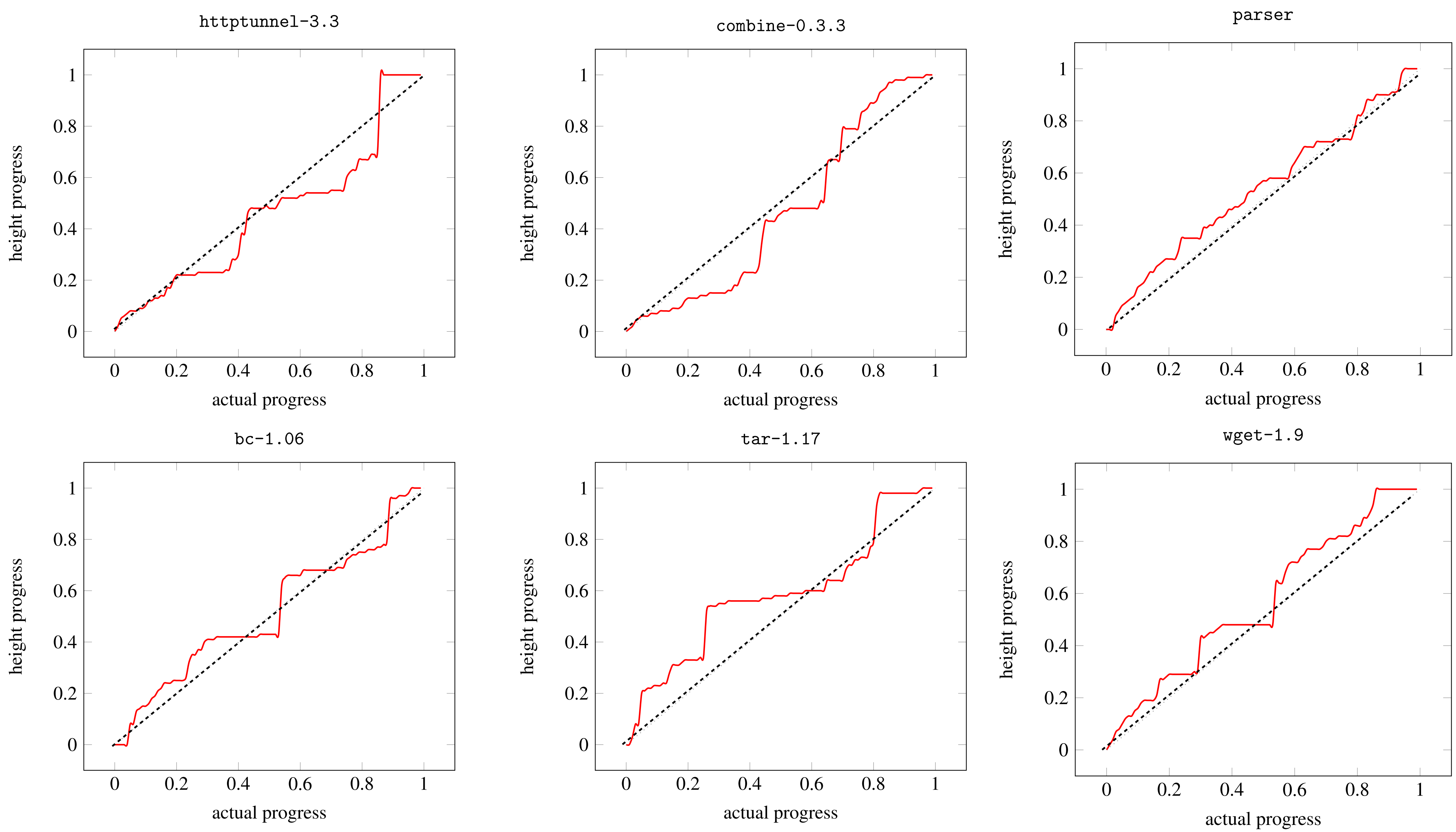
- X : actual progress, Y : estimated progress



Result (Pointer)



Result (Octagon)

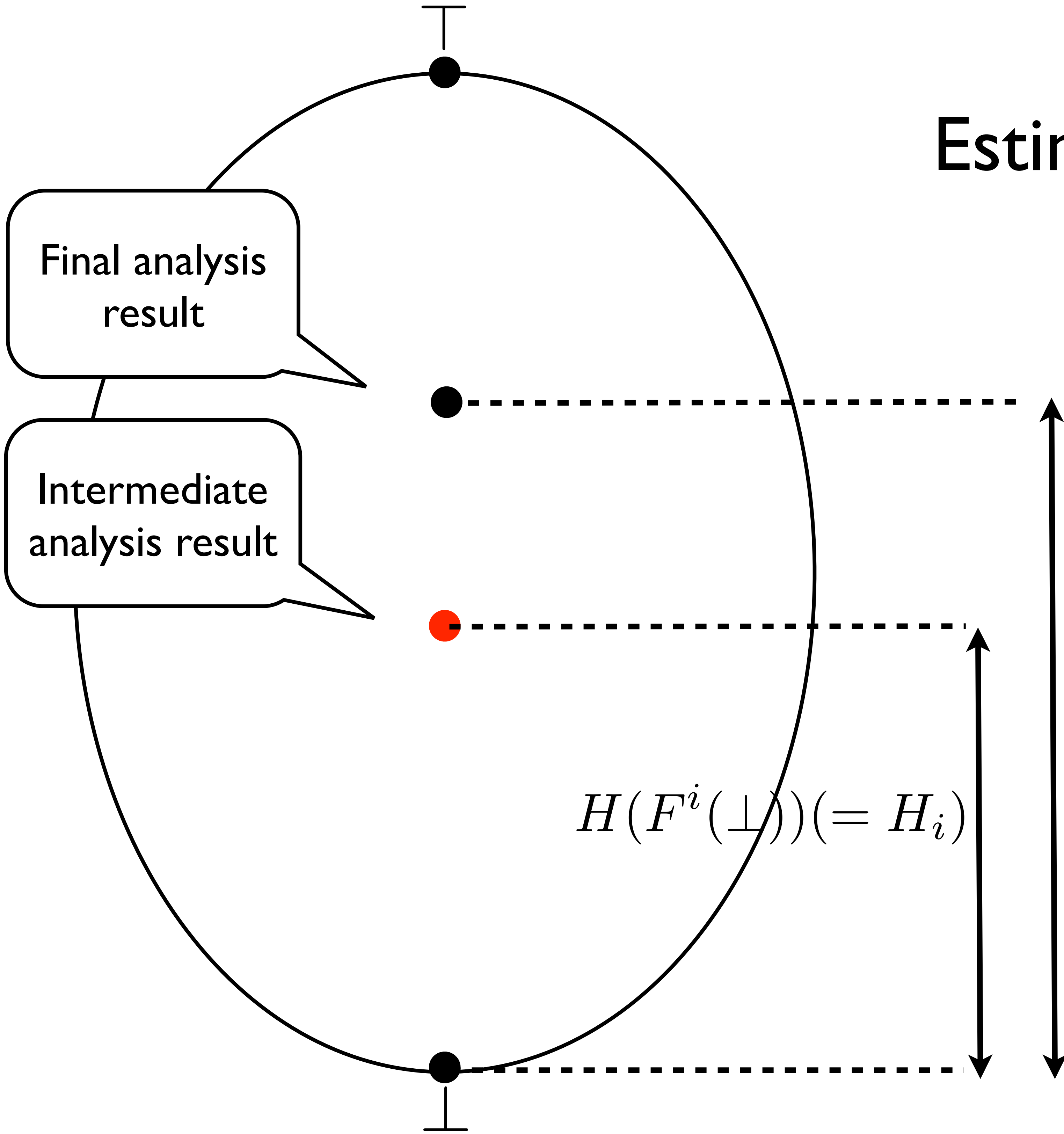


Result

- Time overhead for progress estimations (on 8 GNU programs)
    - Interval analysis : **3.8%**
    - Pointer analysis : **7.3%**
    - Octagon analysis : **36.6%** (prototypical)

Our Approach

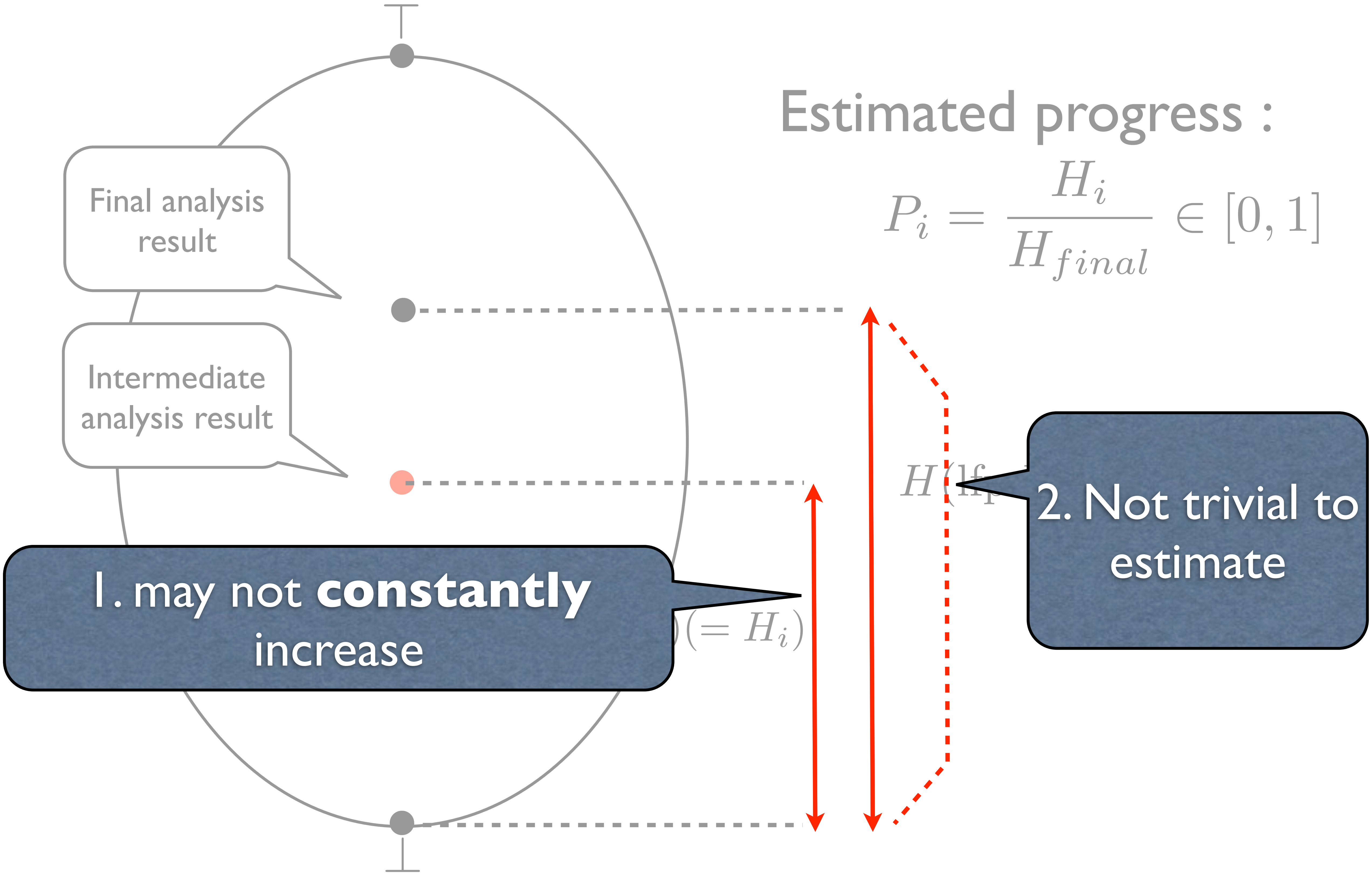
A Naive Approach



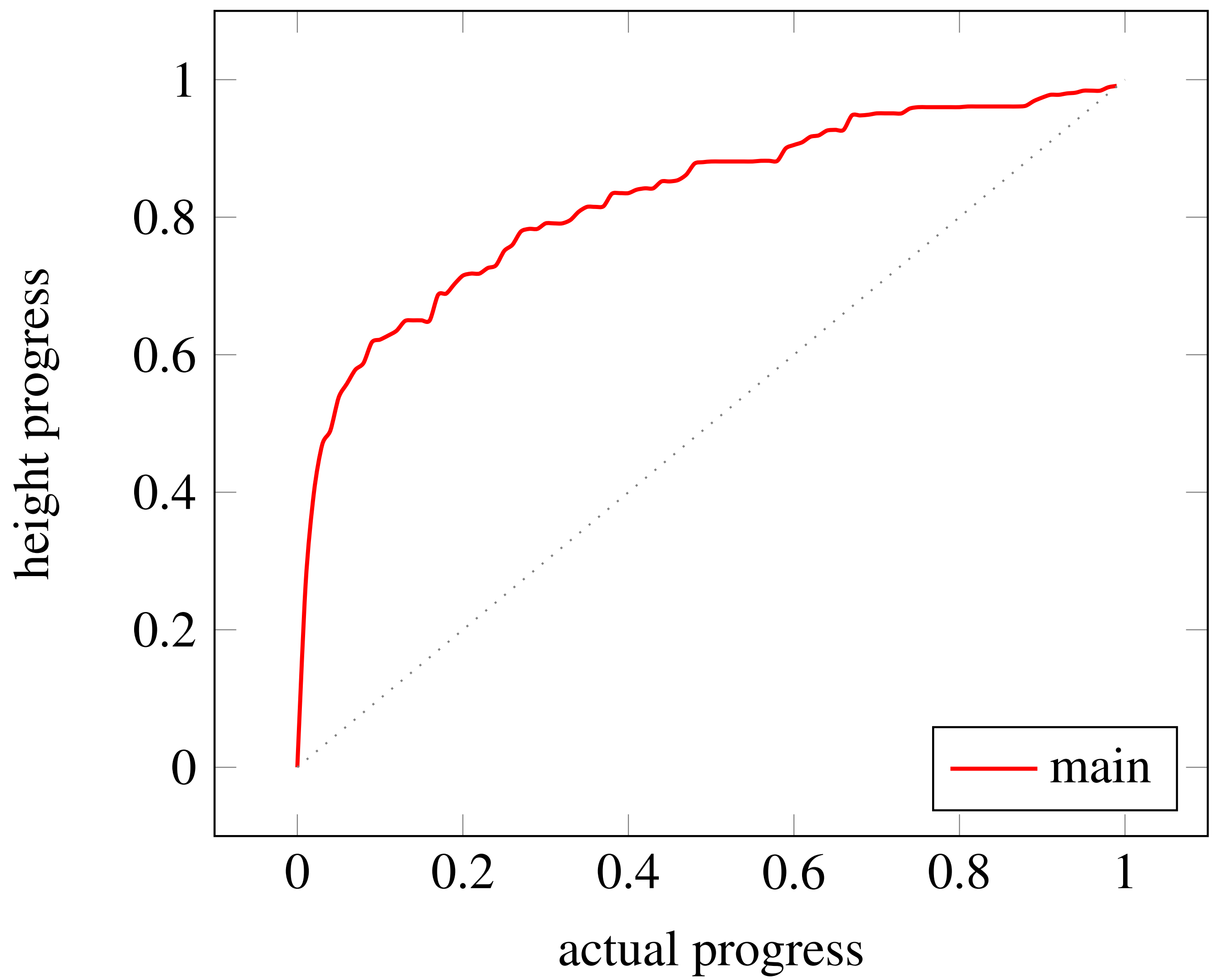
Estimated progress :

$$P_i = \frac{H_i}{H_{final}} \in [0, 1]$$

Problems of the Naive Approach



Problems of the Naive Approach



sendmail-8.14.6 (interval analysis)

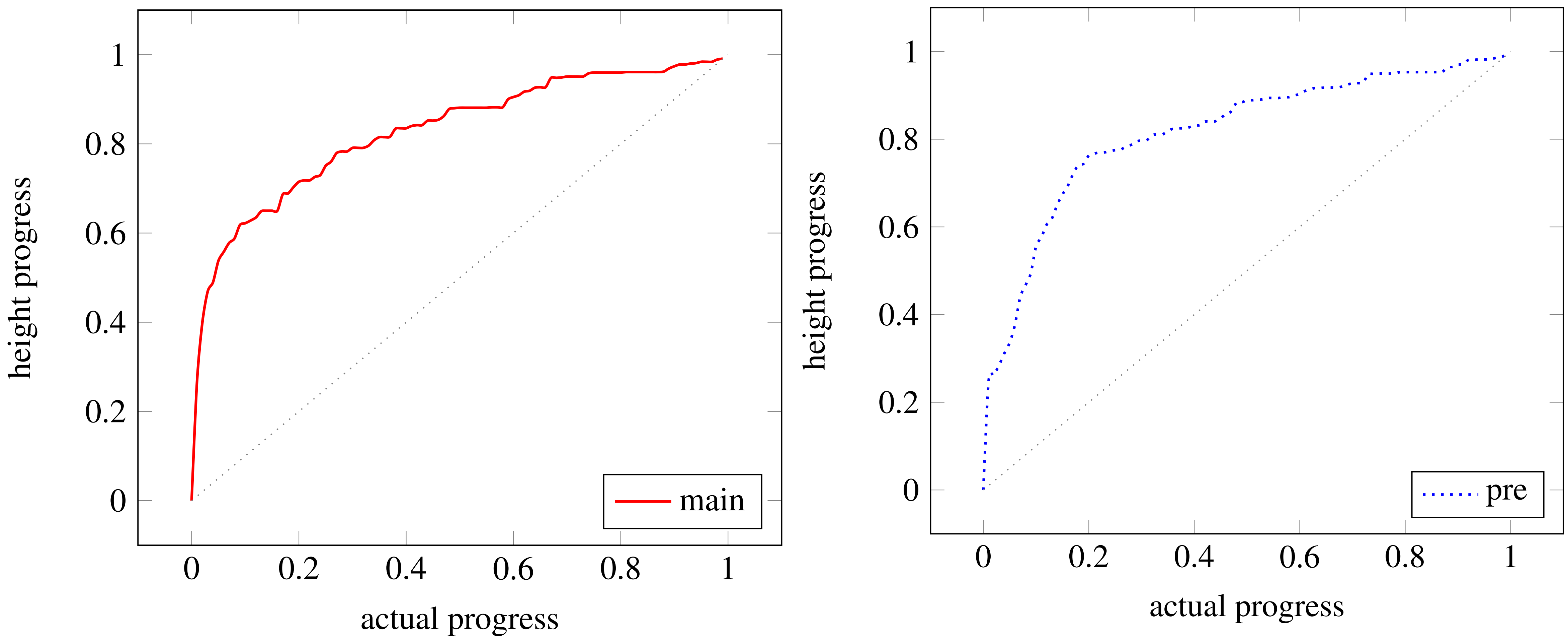
Our Solution

- normalize the height progress

$$\bar{P}_i = \text{normalize}(P_i) = \text{normalize}(\frac{H_i}{H^{\sharp}_{final}})$$

- we can predict 'normalize' by using a less precise, but cheaper <u>pre-analysis.</u>

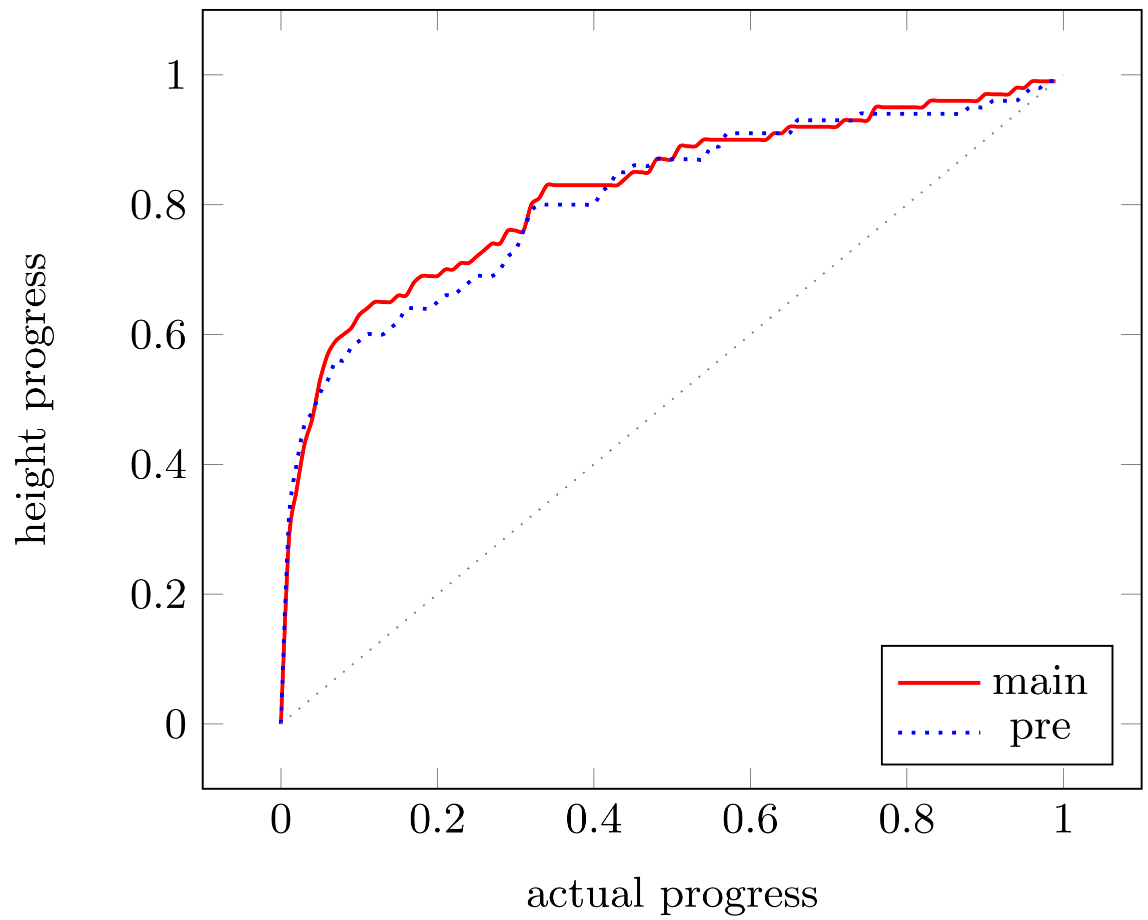
Similar Height-progresses



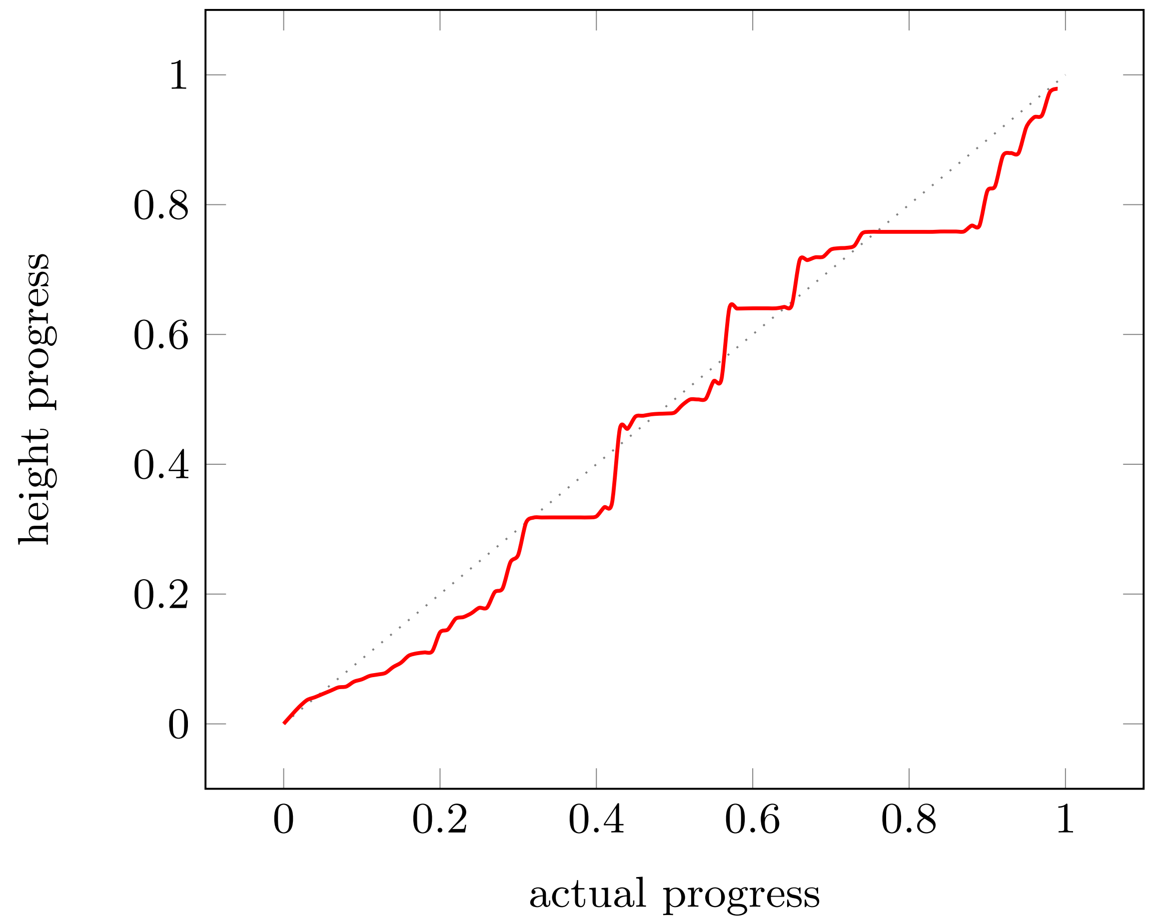
sendmail-8.14.6 (interval analysis)

The pre-analysis takes only 6.6% of the main analysis time.

Normalized Height-progress



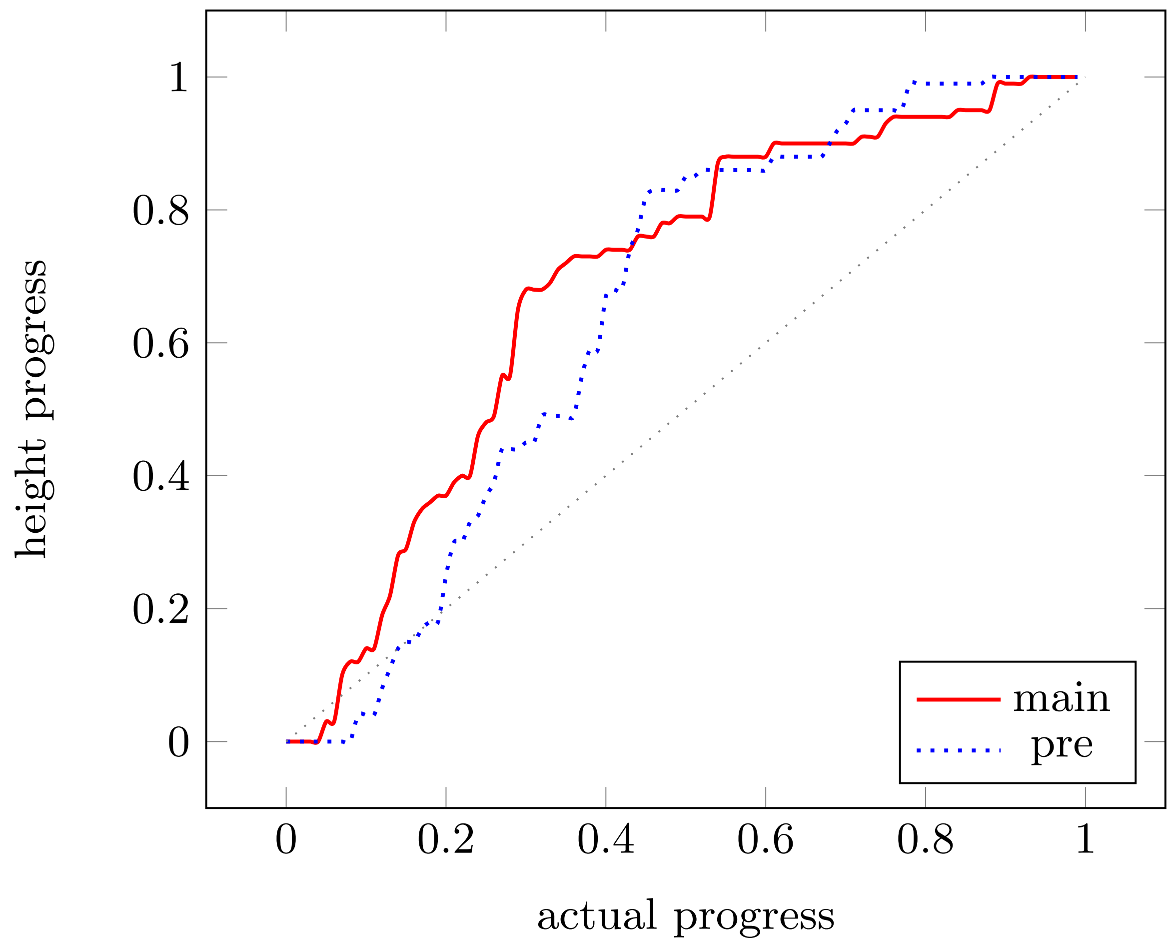
(a) original height-progress



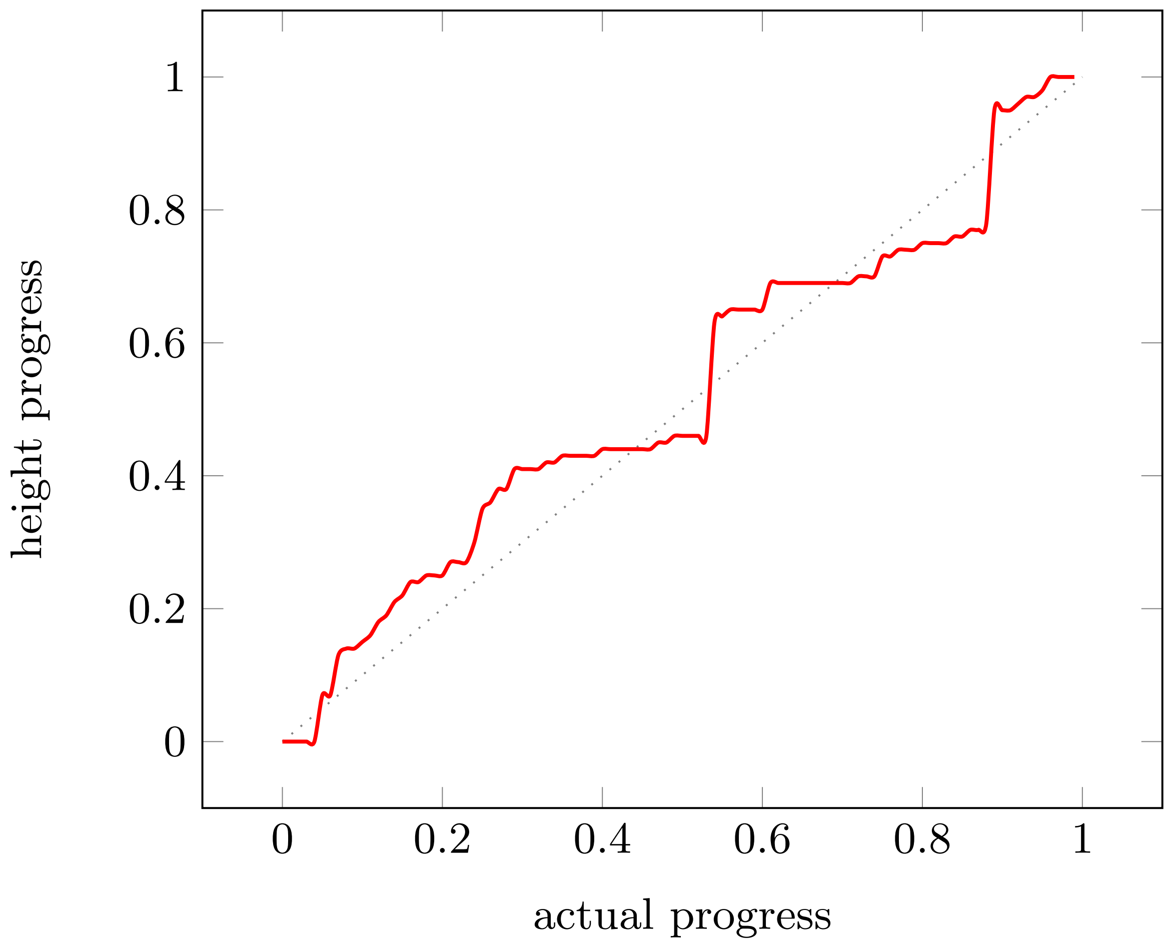
(b) normalized height-progress

sendmail-8.14.6 (interval analysis)

Normalized Height-progress



(a) original height-progress



(b) normalized height-progress

wget-1.9 (octagon analysis)

Our Hypothesis

- If the pre-analysis is semantically related with the main analysis, the pre-analysis's height-progress behavior is similar to that of the main analysis.

The Normalization

- Step 1 : Design a pre-analysis as a further abstraction of the main analysis.

$$\mathbb{D} \underset{\alpha}{\overset{\gamma}{\leftrightarrows}} \mathbb{D}^\sharp$$

$$F^\sharp : \mathbb{D}^\sharp \to \mathbb{D}^\sharp \qquad\qquad \alpha \circ F \sqsubseteq F^\sharp \circ \alpha$$

$$\bigsqcup_{i \in \mathbb{N}} F^{\sharp^i}(\perp^\sharp) = F^{\sharp^0}(\perp^\sharp) \sqcup F^{\sharp^1}(\perp^\sharp) \sqcup F^{\sharp^2}(\perp^\sharp) \sqcup \cdots$$

The Normalization

- Step 2 : Profile the following data during the pre-analysis (suppose the pre-analysis stabilizes in *m* steps)

$$(\frac{H_0^\sharp}{H_m^\sharp}, \frac{0}{m}), \quad (\frac{H_1^\sharp}{H_m^\sharp}, \frac{1}{m}), \quad \cdots, \quad (\frac{H_i^\sharp}{H_m^\sharp}, \frac{i}{m}), \quad \cdots, \quad (\frac{H_m^\sharp}{H_m^\sharp}, \frac{m}{m})$$

where $H_i^\sharp = \mathsf{H}(\gamma(F^{\sharp^i}(\perp^\sharp)))$.

The Normalization

- Step 3 : Generalize the profiled data (via techniques such as interpolation or regression), and obtain a normalization function

$$\text{normalize} : [0, 1] \to [0, 1]$$

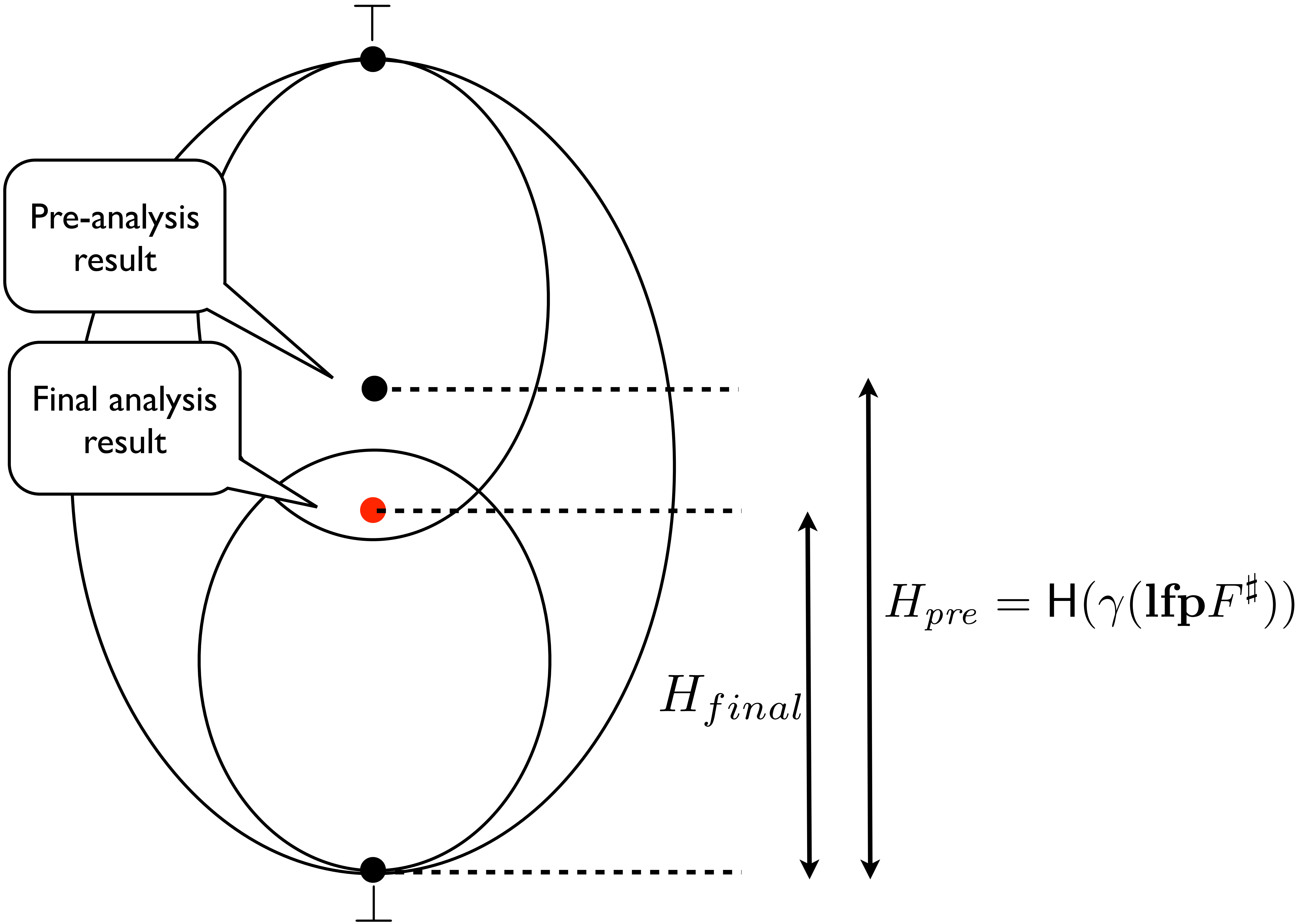
- We use linear regression.

Final Height Estimation

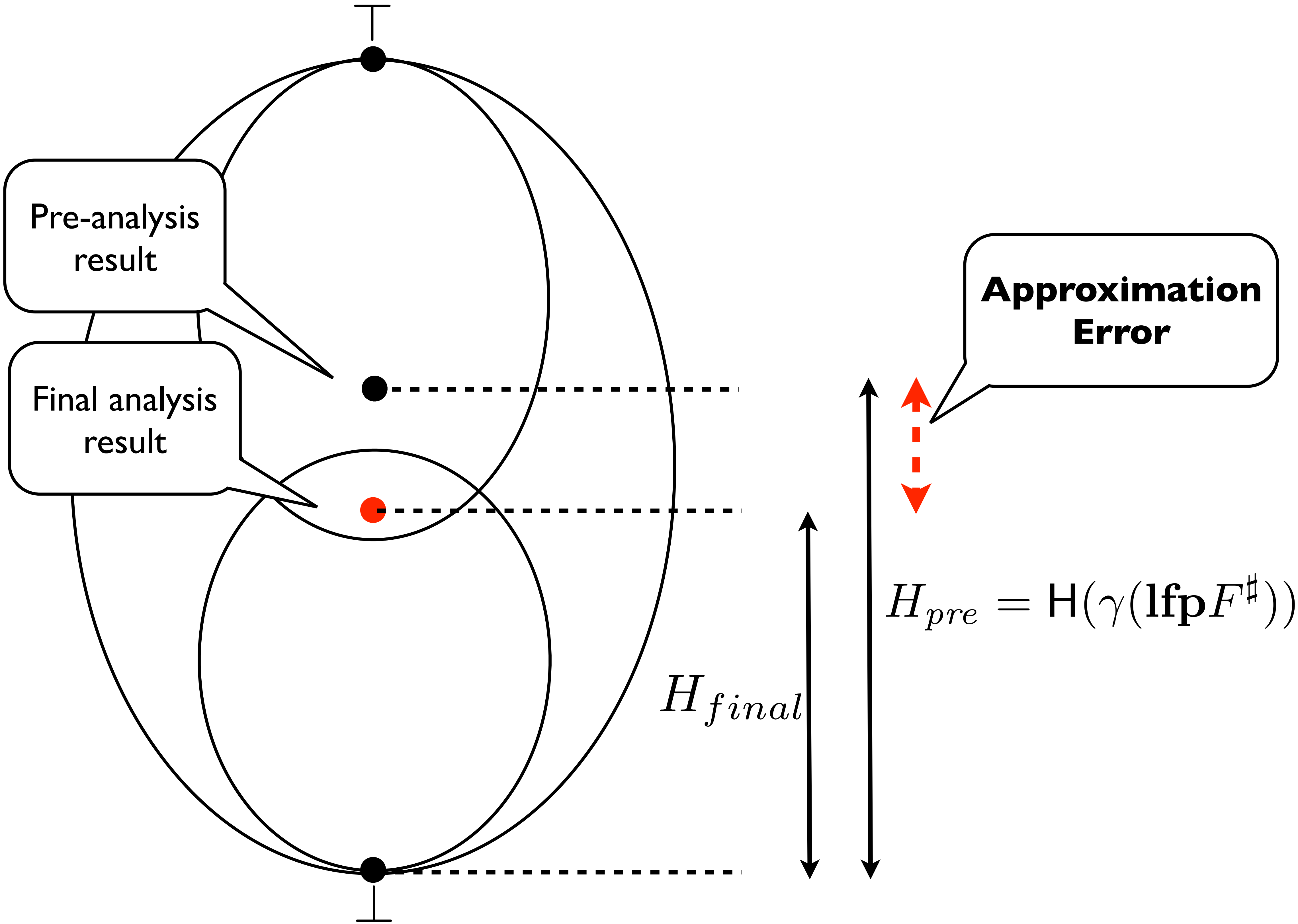
- We use pre-analysis result to estimate the final height.

$$H_{pre} = \mathsf{H}(\gamma(\mathbf{lfp} F^{\sharp}))$$

Final Height Estimation



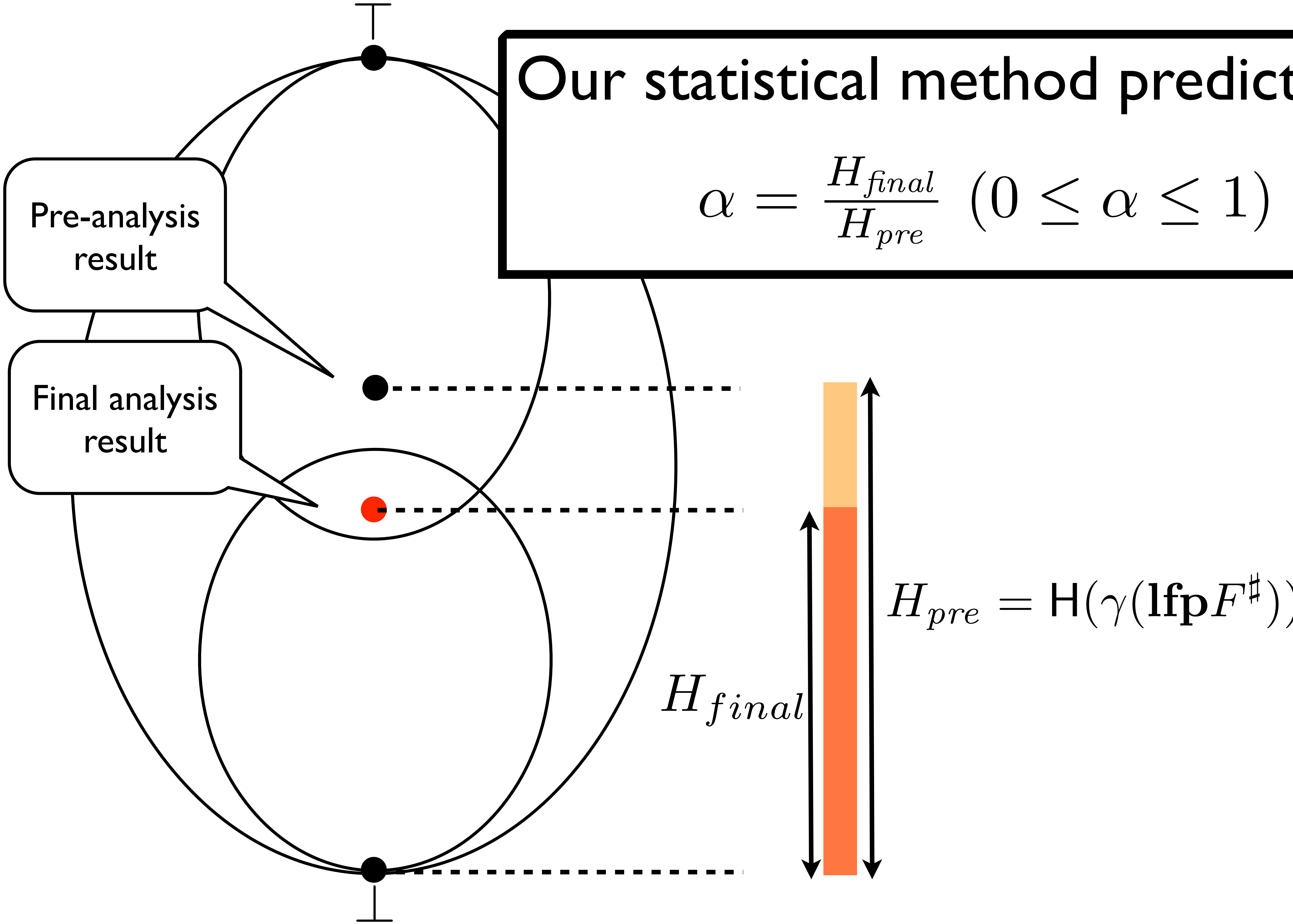
Final Height Estimation



Final Height Estimation

Our statistical method predicts

$$\alpha = \frac{H_{final}}{H_{pre}} \ (0 \leq \alpha \leq 1)$$



Final Height Estimation

- Using ridge linear regression, and **254** programs (GNU, and linux packages)
- 8 syntactic features, 6 semantic features
- Evaluation (3-fold cross validation)
  - Interval : 0.06 mean absolute err. (0.007 std dev.)
  - Pointer : 0.05 (0.001 std dev.)

Details

Progress Estimation Details

- A class of static analyses we consider
- Pre-analysis design
- Precise estimation of a final height

Static Analysis

- Program : $\langle \mathbb{C}, \hookrightarrow \rangle$
- Abstract Domain : program points to abstract states: $\mathbb{D} = \mathbb{C} \to \mathbb{S}$
- Abstract State : abstract locations to abstract values: $\mathbb{S} = \mathbb{L} \to \mathbb{V}$
- Abstract semantic function: $F \in (\mathbb{C} \to \mathbb{S}) \to (\mathbb{C} \to \mathbb{S})$

$$F(X) = \lambda c \in \mathbb{C}. f_c\Big(\bigsqcup_{c' \hookrightarrow c} X(c')\Big)$$

where $f_c \in \mathbb{S} \to \mathbb{S}$ is the transfer function for control point c.

Static Analysis

- Fixpoint computation with widening : If abstract value domain is of infinite height, a widening operator is applied at a set of widening points $\mathbb{W} \subseteq \mathbb{C}$

- In our case, all loop headers are widening points.

Progress Estimation Details

- A class of static analyses we consider
- Pre-analysis design
- Precise estimation of a final height

Pre-analysis

- **Partially flow-sensitive** version of the main analysis (the main analysis is fully flow-sensitive)
- Our pre-analysis only distinguishes program points around loop headers, i.e., widening points
  - as widening increases lattice height significantly.

Partially flow-sensitive analysis

- Abstract domain

$$\mathbb{C} \to \mathbb{S} \xleftarrow[\alpha]{\gamma} \Delta \to \mathbb{S} \qquad \Delta = \Phi \cup \{\bullet\} \qquad \gamma(X) = \lambda c.\ X(\delta(c))$$

- Distinguishable program points Φ

$$\Phi = \{c \in \mathbb{C} \mid w \in \mathbb{W}\ \wedge\ c \hookrightarrow^{depth} w\}$$

adjustable parameter ∈[0,∞]
default : 1

- Semantic function

$$F^\sharp(X) = \lambda i \in \Delta.\ (\bigsqcup_{c \in \delta^{-1}(i)} f_c(\bigsqcup_{c' \hookrightarrow c} X(\delta(c'))) \qquad \delta^{-1}(i) = \{c \in \mathbb{C} \mid \delta(c) = i\}$$

$$\delta(c) = \begin{cases} c & c \in \Phi \\ \bullet & c \notin \Phi \end{cases}$$

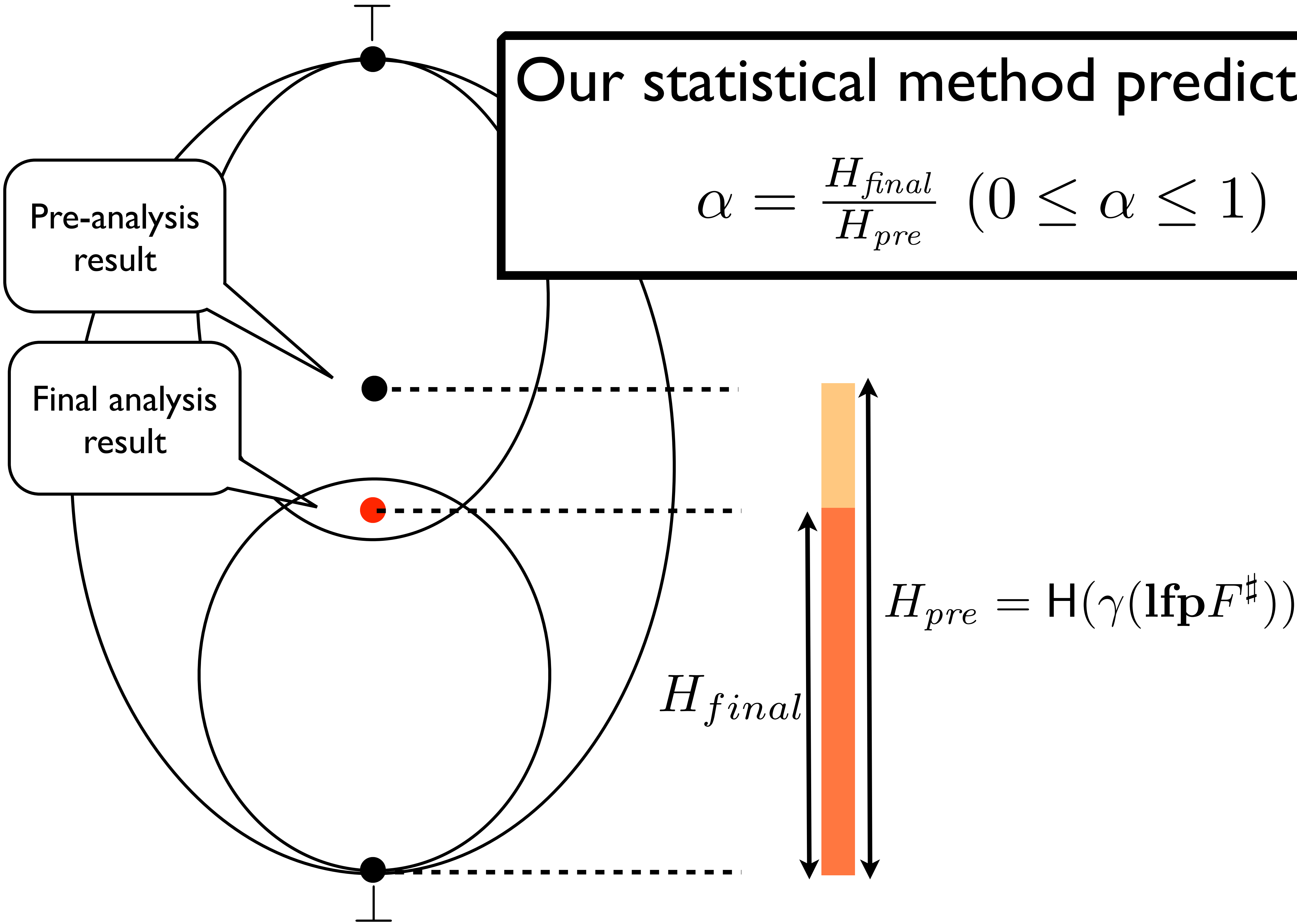
Progress Estimation Details

- A class of static analyses we consider
- Pre-analysis design
- Precise estimation of a final height

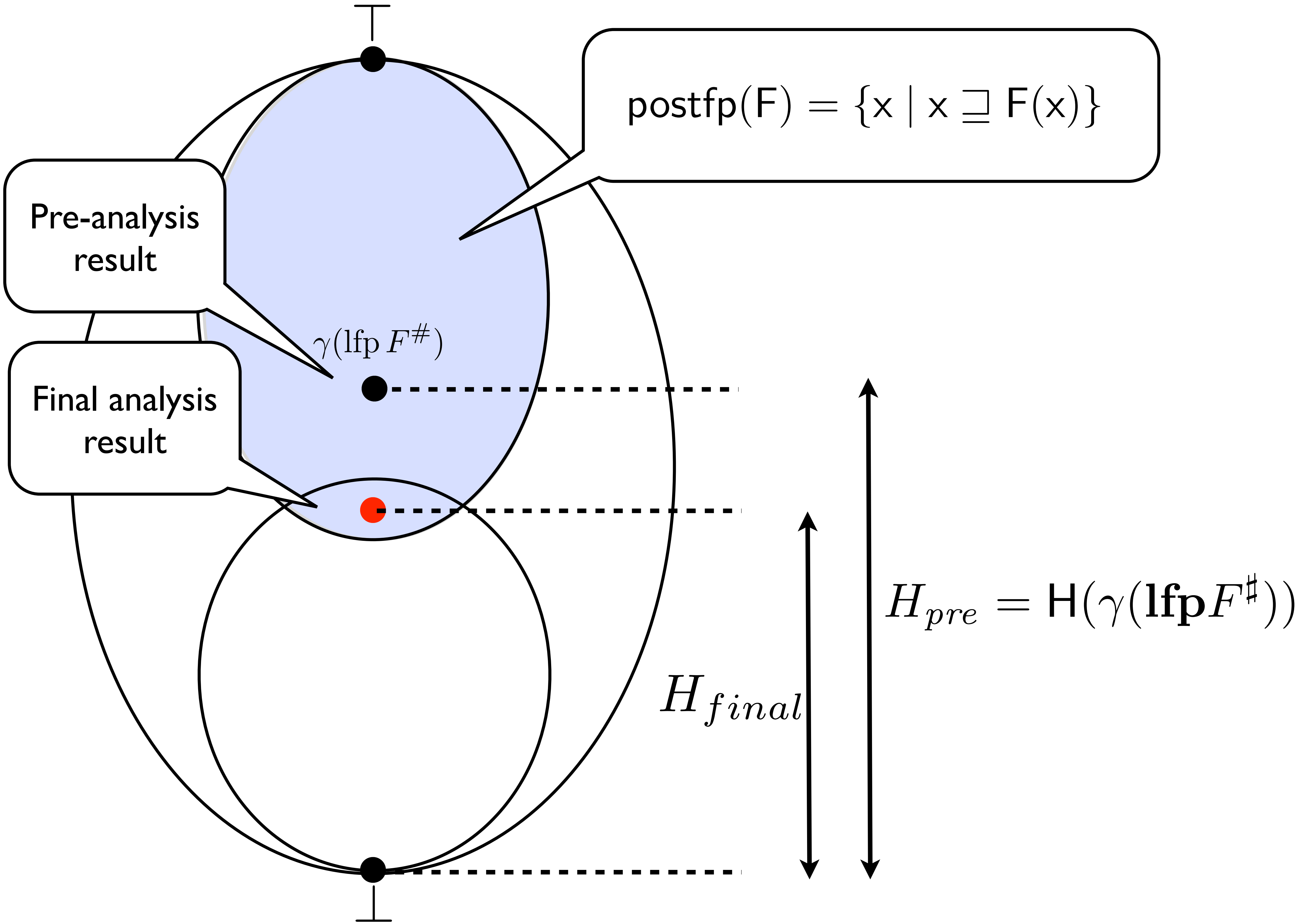
Final Height Estimation

Our statistical method predicts

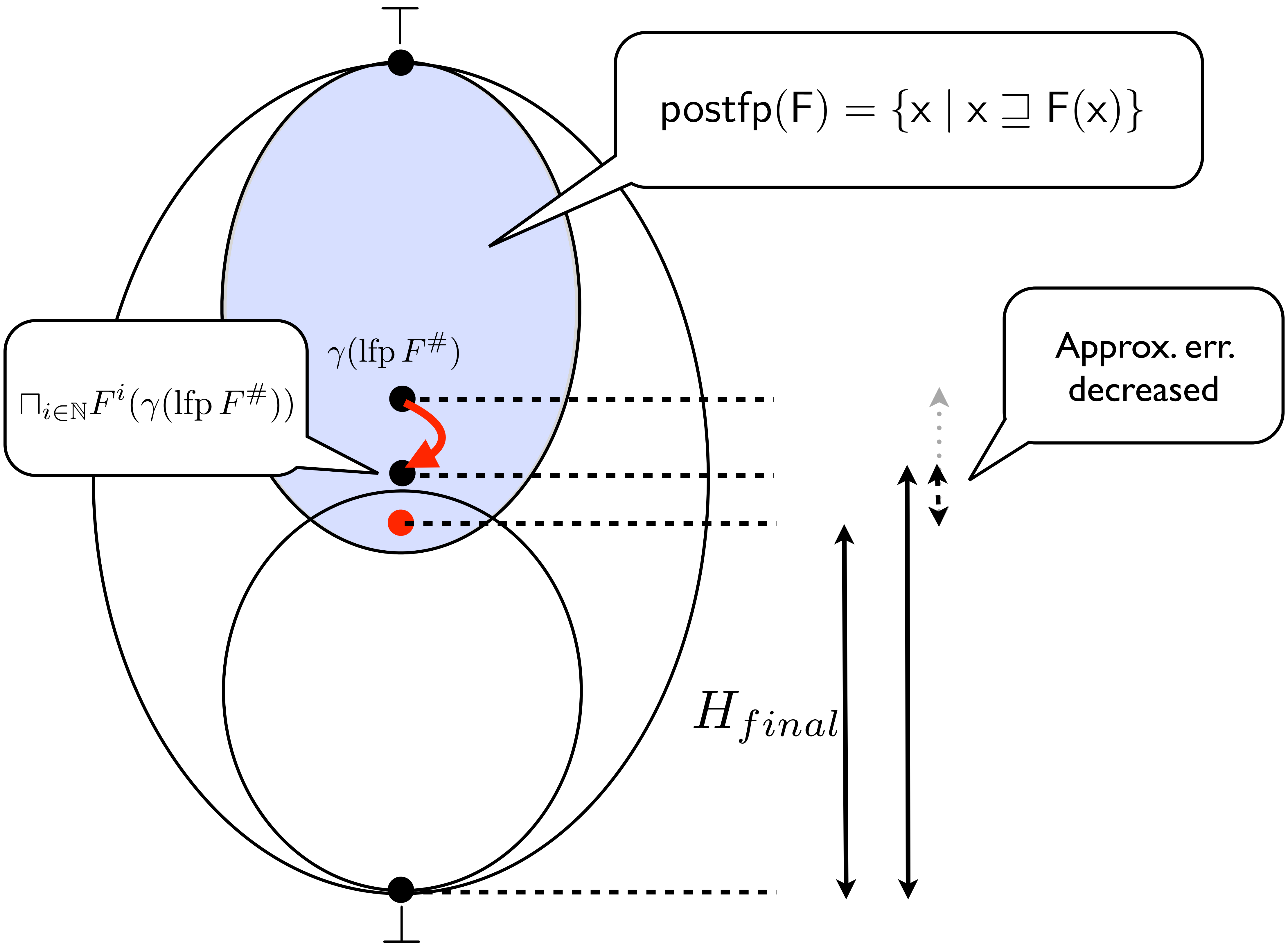
$$\alpha = \frac{H_{final}}{H_{pre}} \quad (0 \leq \alpha \leq 1)$$



Final Height Estimation



Final Height Estimation



Final Height Estimation

- However, the refinement requires significant time overhead.
- We use simple indicators that show possibility of the refinement to avoid extra cost.
  - Height decrease when semantic function F is applied once.

Feature Vector

Table 1. The feature vector used by linear regression to construct prediction models

Category	Feature
Inter-procedural (syntactic)	# function calls in the program
	# functions in recursive call cycles
	# undefined library function calls
Loop-related (syntactic)	the maximum loop size
	the average loop sizes
	the standard deviation of loop sizes
	the standard deviation of depths of loops
	# loopheads
Numerical analysis (semantic)	# bounded intervals in the pre-analysis result
	# unbounded intervals in the pre-analysis result
Pointer analysis (semantic)	# points-to sets of cardinality over 4 in the pre-analysis result
	# points-to sets of cardinality under 4 in the pre-analysis result
Post-fixpoint (semantic)	# program points where applying the transfer function once improves the precision
	height decrease when transfer function is applied once

Evaluation Metric

- Our progress bar : $\bar{P}_i^\sharp = \text{normalize}(\frac{H_i}{\alpha \cdot H_{pre}})$

- We quantifiably measure quality of estimation
  (n : total iteraion, i : current iteration, best : 1)

$$Linearity(\bar{P}_i^{\#}) = 1 - \frac{\sum_{1 \le i \le n}(\frac{i}{n} - \bar{P}_i^\sharp)^2}{\sum_{1 \le i \le n}(\frac{i}{n} - \frac{n+1}{2n})^2}$$

Interval Analysis

Program	LOC	Time(s) Main	Time(s) Pre	Linearity	Overhead	Height-Approx.
bison-1.875	38841	3.66	0.91	0.73	24.86%	1.03
screen-4.0.2	44745	40.04	2.37	0.86	5.92%	0.96
lighttpd-1.4.25	56518	27.30	1.21	0.89	4.43%	0.92
a2ps-4.14	64590	32.05	11.26	0.51	35.13%	1.06
gnu-cobol-1.1	67404	413.54	99.33	0.54	24.02%	0.91
gnugo	87575	1541.35	7.35	0.89	0.48%	1.12
bash-2.05	102406	16.55	2.26	0.80	13.66%	0.93
sendmail-8.14.6	136146	1348.97	5.81	0.69	0.43%	0.93
TOTAL	686380	3423.46	130.5	**0.74**	**3.81%**	**Err : 0.07**

Pointer Analysis

Program	LOC	Time(s) Main	Time(s) Pre	Linearity	Overhead	Height-Approx.
screen-4.0.2	44745	15.89	1.56	0.90	9.82%	0.98
lighttpd	56518	11.54	0.87	0.76	7.54%	1.03
a2ps-4.14	64590	10.06	3.48	0.65	34.59%	1.04
gnu-cobol-1.1	67404	32.27	12.22	0.91	37.87%	1.03
gnugo	87575	217.77	3.88	0.64	1.78%	0.97
bash-2.05	102406	3.68	0.78	0.56	21.20%	1.04
proftpd-1.3.2	126996	74.64	11.14	0.82	14.92%	1.03
sendmail-8.14.6	136146	145.62	3.15	0.58	2.16%	0.98
TOTAL	686380	511.47	37.08	**0.73**	**7.25%**	**Err : 0.03**

Octagon Analysis

Program	LOC	Time(s) Main	Time(s) Pre	Time(s) Linearity	Overhead
httptunnel-3.3	6174	49.5	8.2	0.91	16.6%
combine-0.3.3	11472	478.2	16	0.89	3.4%
bc-1.06	14288	63.9	43.8	0.96	68.6%
tar-1.17	18336	977.0	73.1	0.82	7.5%
parser	18923	190.1	104.8	0.97	55.1%
wget-1.9	35018	3895.36	1823.15	0.92	46.8%
TOTAL	69193	5654.0	2069.49	**0.91**	**36.6%**

Precision-Overhead Tradeoff

- More finer partitioned pre-analysis → more precise progress estimation

Program	**Linearity change**	**Overhead change**
bash-2.05 (pointer)	0.56 → 0.70	21.2% → 37.5%
sendmail-8.14.6 (interval)	0.69 → 0.95	0.4% → 18.4%

Conclusion

- For the first time, we propose a technique for estimating static analysis progress.
- Our method combines a semantic-based pre-analysis with machine learning.
- We show its applicability on a suit of real C benchmarks.

Backup

Height Function

$$\mathsf{H} : (\mathbb{C} \to \mathbb{S}) \to \mathbb{N}$$

$$\mathsf{H}(X) = \sum_{c \in \mathbb{C}} \sum_{l \in \mathbb{L}} \mathsf{h}(X(c)(l))$$

- Interval

$$\mathsf{h}(\perp) = 0$$

$$\mathsf{h}([a,b]) = \begin{cases} 1 & a = b \ \wedge \ a,b \in \mathbb{Z} \\ 2 & a < b \ \wedge \ a,b \in \mathbb{Z} \\ 3 & a \in \mathbb{Z} \ \wedge \ b = +\infty \\ 3 & a = -\infty \ \wedge \ b \in \mathbb{Z} \\ 4 & a = -\infty \ \wedge \ b = +\infty \end{cases}$$

- Pointer

$$\mathsf{h}(S) = \begin{cases} 4 & |S| \geq 4 \\ |S| & \textit{otherwise} \end{cases}$$

- Octagon : we reuse interval's height function.