EXPLORING THE EFFECTIVENESS OF USING PROMISING SUBPROGRAMS IN PROGRAM SYNTHESIS WITH BUDGETED SEARCH

Author:
Alexander Jelev | A.Jelev@student.tudelft.nl

Supervisor
Sebastijan Dumančić | S.Dumancic@tudelft.nl

BACKGROUND

Program Synthesis

- Generate programs from examples
- Search over a program space

Herb.jl [1]

FrAngel

-Implementation
- Special-case similarity

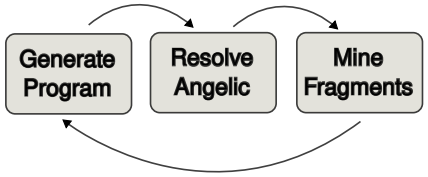


Figure 1: Block diagram of FrAngel implementation

RESEARCH QUESTION

How do we find frequently occurring subprograms in different attempts and effectively leverage them, following the idea of FrAngel?

Experimental Questions

- Does budgeted search with useful subprograms improve the effectiveness of the synthesis task?
- Does budgeted search with useful subprograms find better intermediate solutions than a standard synthesis task?

METHODS

Budgeted Search

- Budget is number of programs enumerated
- Repeat:
  - synthesize and evaluate programs
  - select results
  - make updates
  - check stopping criteria

Useful Subprograms Implementation

- Update Grammar step corresponds to FrAngel's Mine Fragments step
- Only includes full subprograms in grammar updates

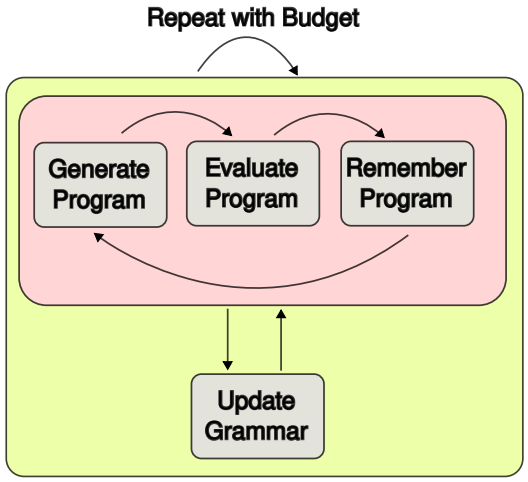


Figure 2: Block diagram of budgeted useful subprograms implementation

EXPERIMENTS

Approach	Total Benchmarks Evaluated
Sketches	204
Useful Subprograms	148
Proabilistic Grammar	204
	Optimal Solutions
Sketches	51
Useful Subprograms	34
Proabilistic Grammar	33

Table: Summary of total benchmarks solved optimally by each budgeted search approach

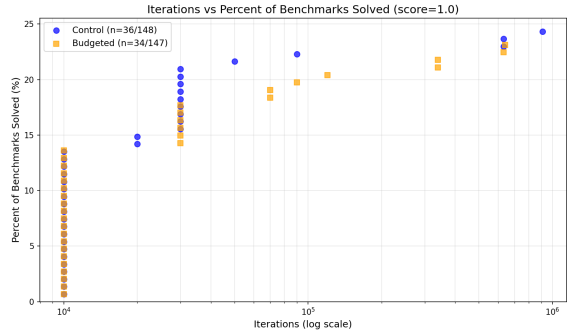
DelftBlue Setup

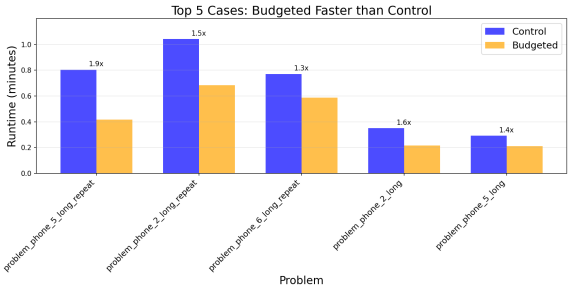
- Memory partition
- 10 CPUs, 15G each
- OpenMPI (10 processes)

Julia Implementation

- SizeBasedBottomUpIterator
- interpret_sygus
- PBE_SLIA_Track_2019 Benchmarks (HerbSearch)
- 30 mins time limit per benchmark

Experimental Results





CONCLUSIONS AND FUTURE WORK

- Useful Subprograms implementation was mostly beaten by control runs
- This could be due to benchmark test set not exposing special cases

Future Work:
- Test implementation with more benchmarks
- Avoid reconstructing iterator after grammar updates

REFERENCES

[1] PONY lab - Algorithmics Group - TU Delft, "Herb.jl - A program synthesis library written in Julia," herb-ai.github.io, Accessed: Jan. 26, 2026. [Online]. Available: https://herb-ai.github.io/

TU Delft