Progressions User Guide

Version 1.0

Derrick Stolee
University of Nebraska-Lincoln
s-dstolee1@math.unl.edu

February 27, 2012

Abstract

The Progressions software searches for colorings of $\{1, \ldots, n\}$ that avoid certain types of monochromatic progressions: quasi-arithemtic progressions and pseudo-arithmetic progressions.

1 Acquiring *Progressions*

The latest version of *Progressions* and its documentation is available online as part of the *SearchLib* collection at the address

`http://www.math.unl.edu/~s-dstolee1/SearchLib/`

Progressions is made available open-source under the GPL 3.0 license.

To complile *Progressions*, use a terminal to access the `Progressions/src/` folder and type `make`. The executables will be placed in `Progressions/bin/`

1.1 Acquiring Necessary Libraries

There are two *SearchLib* projects used by *Progressions*.

1. *TreeSearch* is a project in *SearchLib* that abstracts the structure of a backtrack search in order to allow for parallelization. *TreeSearch* is available on the same web site as *Progressions*. Consult the *TreeSearch* documentation for details about the arguments and execution processes.
2. *Utilities* is a project in *SearchLib* containing useful objects and functions necessary by other projects in *SearchLib*. *Utilities* is available on the same web site as *Progressions*.

1.2 Full Directory Structure

For proper compilation, place the different dependencies in the following directory structure:

- `SearchLib/` – The *SearchLib* collection.
    - `Progressions/` – The *Progressions* project.
        - `bin/` – The final binaries are placed here.
        - `docs/` – This folder contains documentation.
        - `src/` – Contains source code. Compilation occurs here.

- `TreeSearch/` – A support project from *SearchLib*.
- `Utilities/` – A support project from *SearchLib*.
  - `src/` – Type `make` in this directory to compile the *Utilities* project.

2 Execution

The main executable is `progressions.exe`.

2.1 Progessions-Specific Arguments

- `--mode [quasi|pseudo]` — Select which type of progression to avoid: Quasi-Arithmetic or Pseudo-Arithmetic.
- `-R #` — The number of colors to use (default: 2).
- `-n #` — The minimum length of a good coloring to report. Will be used by constraint propagation to prune the search space.
- `-N #` — The maximum number of elements to color. Propagation and coloring will not extend beyond this value.
- `-K #` — The length of the progressions.
- `-D #` — The diameter of the progressions.
- `-I #` — The diameter as $d = k - i$. (**Warning:** Must follow the argument of `-K #`).
- `--skew-symmetric` — If present, the colorings will be restricted to skew-symmetric colorings. In this case, the colorings span $\{-n, \dots, -1, 0, 1, \dots, n-1\}$.
- `--backward [on|off]` — Specify if the backward propagation should be enabled.
- `--forward [on|off]` — Specify if the forward propagation should be enabled. If enabled, the backward propagation will be enabled as well.

2.2 TreeSearch-Specific Arguments

- `-k #` — The killtime: How many seconds before halting the process and reporting a partial job.
- `-m #` — The maximum depth: the maximum number of steps to go before halting (or in generation mode, a new job is written at this depth).
- `run` — Run mode: The input jobs are run until finished or the killtime is reached.
- `generate` — Generation mode: The input jobs are run and new jobs are listed when reaching the maximum depth.
- `--maxjobs #` — The maximum number of jobs to generate before halting with a partial job (default: 1000).
- `--maxsols #` — The maximum number of solutions to output before halting with a partial job (default: 100).

References

[1] A. S. Jobson, A. E. Kézdy, D. Stolee, A new variant of van der Waerden numbers, *in preparation*.

[2] D. Stolee, TreeSearch user guide, available at http://www.github.com/derrickstolee/TreeSearch/ 2011.