Evolving $\sqrt{x}$ into $1/x$
via software data maintenance
GI @ GECCO 2020 workshop. doi:10.1145/3377929.3398110

W. B. Langdon
and
Oliver Krauss
Genetic Improvement via Data

• Different type of Genetic Improvement
• Optimise embedded constants, i.e. data
  1. GI on data to evolve new or better functionality
  2. Same functionality, but better (e.g. use less energy)
• Why
  1. Minimal code changes may be more acceptable?
  2. Easier route into GI: use your favourite optimisation tool on numbers inside program source code.
  • The objective measure is based on how existing program behaves with new data inside it.
  • Eg optimise internal parameters, answer x% better
Why Reciprocal $\frac{1}{x}$

- Example of Genetic Improvement on data
- On hardware with slow division (possibly mote computing IoT) multiplication by GI double precision reciprocal could be faster

$$a \times \text{drcp}(b) \text{ could be quicker than } \frac{a}{b}$$
Genetic Improvement via Data

• New functionality via data changes
  – RNAfold better predictions [EuroGP 2018]
  – New maths functions, e.g. convert $\sqrt{x}$ into $1/x$

• In future can you improve software (e.g. faster) whilst keeping old behaviour?

RNA molecule. Example of more accurate RNAfold output
Maintaining Embedded Constants

- **EuroGP 2018**
  - RNAfold 7000 lines of code 50000 numbers
  - On average better predictions of RNA folding.
  - Shipped since 2018 (release 2.4.7)

- Evolution plus *manual changes* to open source **GNU C** library double precision sqrt gives:
  - cube root
  - $\log_2$
  - $\text{invsqrt } \frac{1}{\sqrt{x}}$

  - division less division $1/x$
Use **CMA-ES** to convert sqrt into $1/x$

By updating table of 512 floats used by **PowerPC** code.

**IEEE 754 Double Precision**

---

**C double precision square root function**

![Graph showing C double precision square root function](image)
GNU C library sqrt $\sqrt{x}$ converted to $1/x$

- sqrt puts normalised input $x$ into 512 bins
- Each bin holds a start point for Newton-Raphson

- Run CMA-ES per bin (total $\geq 512$ times)
  - Seed with square root data
  - Run drcp code with CMA-ES generated value to give $c=\text{drcp}(x)$. Inverting $c$ should give $x$
  - Fitness based on difference $|1/c - x|$
    $x$ takes 3 test values: smallest, mid, max in bin

- Usually all differences smallest possible
- else run CMA-ES again
CMA-ES finds good values for all Newton-Raphson bins
Evolved drcp \( \frac{1}{x} \)

Evolved double reciprocal drcp tested many thousands of times

- It works!
  - Always within DBL_EPSILON \((2^{-52}, 2.2 \times 10^{-16})\)
    - I.e. smallest possible difference still visible in double precision arithmetic
  - Almost always gives best possible double

- Easy. (six seconds)
- You can do better.

W. B. Langdon
Automatic Software Maintenance

• In a world addicted to software, maintenance is the dominant cost of computing.

• Need to keep parameters up to date. E.g.
  – New science, new laws or regulations, new users, new user expectations
  – Change of load, new hardware (e.g. bigger RAM memory), auto port to new devices/phones
  – Search can be fast (total CMA-ES runtime 6 secs)

• Little SBSE research

• Great scope for automation
Summary: Genetic Improvement on Data

• Problem of maintaining data in code ignored
• GI can optimize data in programs
  • Use your own favourite optimiser on parameters numbers embedded in source code, with your objective: battery life, faster, better predictions, etc., etc.

• Rapidly generated maths \( (\text{cbrt}, \log_2, \frac{1}{\sqrt{x}}, \frac{1}{x}) \)
  • Replication package GitHub [Replication_GI_Division_Free_Division]

• Software is not fragile
END

W. B. Langdon YouTube videos
https://www.youtube.com/channel/UChebBvv66dPOcElWk6ht4OA
Genetic Programming

W. B. Langdon
Improving RNAfold parameters

- RNAfold 7100 lines of C source code, 51521 parameters.
- Fitness correlation between prediction and true structure (MCC).
- Post evolution tidy
- 14732 (29%) parameters changed
- Holdout set significant increase in MCC
- Also better than constrained optimisation
- GI parameters [rna_langdon2018.par](https://example.com) shipped with ViennaRNA since 13 Jun 2018
Manual Changes I of 2 (old sqrt)

• Most implementations of square root use hardware support.
• GNU C library glibc 2.29 also includes Newton-Raphson iterative solution
• Trap bad values, e.g. negative
• Normalise double input to 0.5 .. 2.0
• Guaranteed convergence in three steps:
  – Update both estimate of $\sqrt{x}$ and derivative
• Apply square root to exponent, ie divide by 2
Code Changes II (for 1/x)

- Normalise double precision input to 1.0..2
  - Update estimate of $x^{-1}$
  - Use reciprocal of derivative, i.e. $-x^{-2}$, directly
- Apply 1/x to exponent, i.e. negate.
- Could we use code GI to further improve?

- Source code
  
  http://www.cs.ucl.ac.uk/staff/W.Langdon/ftp/gp-code/gi_cbrt.tar.gz
Square root to $\log_2$

Frame work as sqrt to cbrt but

- Derivative known
- CMA-ES one dimension at a time (512 times) very easy

W. B. Langdon 17
The Genetic Programming Bibliography

http://gpbib.cs.ucl.ac.uk/

13569 references, 12000 authors

Make sure it has all of your papers!
E.g. email W.Langdon@cs.ucl.ac.uk or use | Add to It | web link

RSS Support available through the Collection of CS Bibliographies.

Co-authorship community.
Downloads

A personalised list of every author’s GP publications.

blog

Googling GP Bibliography, eg: earthquake site:gpbib.cs.ucl.ac.uk