CREST

Improving SSE Parallel Code with Grow and Graft Genetic Programming

Genetic Improvement 2017, D.R. White *et al*., GECCO companion, pages 1537-1538, Berlin

W. B. Langdon
Department of Computer Science

9.7.2017

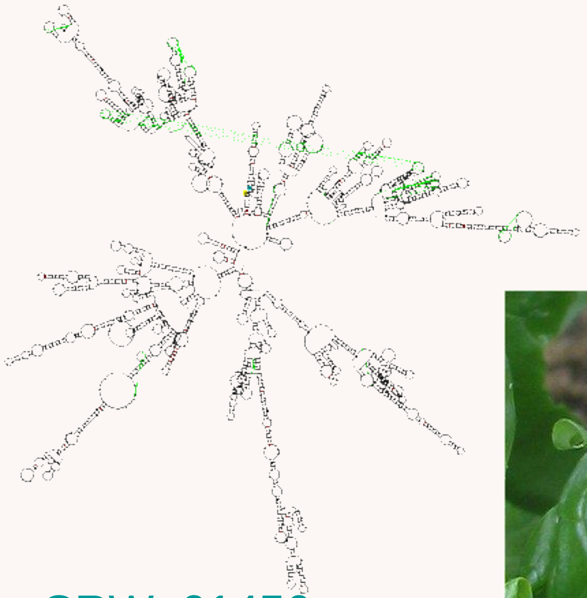
Improving SSE Parallel Code with GGGP

- Background
  - RNAfold, test on real data from RNA_STRAND
  - Manual introduction of Intel SSE code
  - GI on SSE code
- Results
  - 30%
  - GI 1.6% on top
- Manual code in ViennaRNA
  - 2190 downloads since 14 April
  - V2.3.5 needs `./configure --enable-sse`

RNA STRAND

Known structure of 4666 RNA molecules

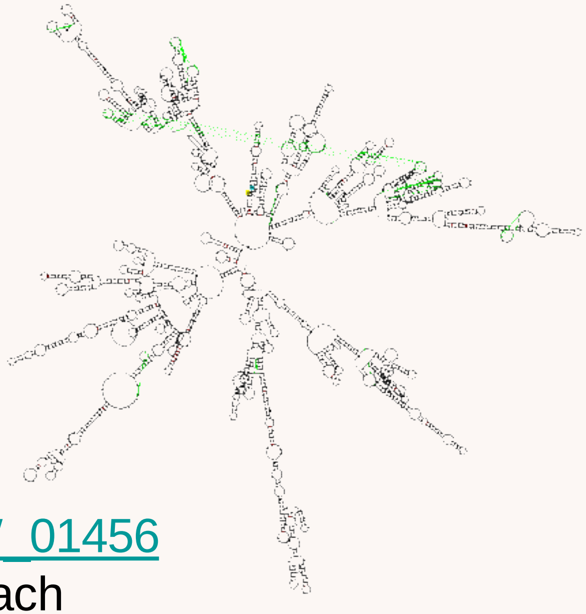
<table>
<tr><td colspan="6"># File CRW_01456.ct</td></tr>
<tr><td colspan="6"># RNA SSTRAND database</td></tr>
<tr><td colspan="6"># External source: Gutell Lab CRW, file name:</td></tr>
<tr><td>1</td><td>U</td><td>0</td><td>2</td><td>2912</td><td>1</td></tr>
<tr><td>2</td><td>U</td><td>1</td><td>3</td><td>2911</td><td>2</td></tr>
<tr><td>3</td><td>C</td><td>2</td><td>4</td><td>2910</td><td>3</td></tr>
<tr><td>…</td><td></td><td></td><td></td><td></td><td></td></tr>
<tr><td>2911</td><td>A</td><td>2910</td><td>2912</td><td>2</td><td>2911</td></tr>
<tr><td>2912</td><td>A</td><td>2911</td><td>2913</td><td>1</td><td>2912</td></tr>
<tr><td>2913</td><td>C</td><td>2912</td><td>2914</td><td>0</td><td>2913</td></tr>
</table>

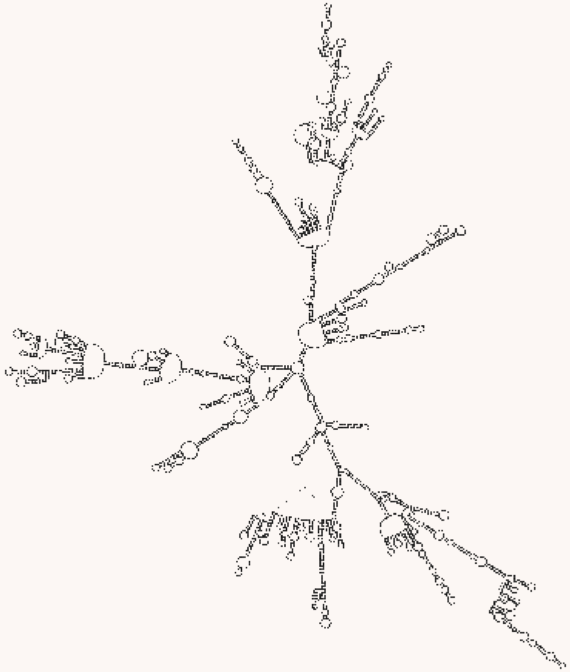


CRW_01456
Spinach
23S Ribosomal RNA
2913 bases

RNAfold

- Part of ViennaRNA package (170000 lines)
- RNAfold 7100 lines .c (i.e. excluding .h)
- State of the art, users include EteRNA





CRW_01456
Spinach
23S Ribosomal RNA
2913 bases

Matthews correlation coefficient MCC 0.634283

RNAfold

- Takes sequence of bases, predicts structure
- Profiled gnu gcov on all of RNA_STRAND
- 4 lines executed billions of times

CRW_01456.fasta

```
> CRW_01456
UUCAAACGAGGAAAGGCUUACGGUGGAUACCUAGGCACCCAGAGACGAGG
AAGGGCGUAUUAAUCGACGAAAUGCUUCGGGGAGUUGAAAAUAAGCAGAG
AUCCGGAGAUUCCCGAAUAGGUCAACCUUUCGAACUUCUGCUGAAUCCAU
```

RNAfold output

```
./RNAfold < CRW_01456.fasta
.........((...((((((((((....)))..(((((((((...(
```

```
for (decomp=INF, k=i + 1 + turn; k<=stop; k+
  if((fmi[k] != INF ) && (fm[k1j] != INF)){
    en = fmi[k] + fm[k1j];
    decomp = MIN2(decomp, en);
  }
}
```

Original code

For loop executed 1..3000 times,
average run time 400 nanoseconds

Manual introduction of SSE code

- ≈3 mandays to generate SSE code
  - Use C interface, e.g. `_mm_add_epi32` add 4 int
  - Use `min` and `shuffle` to take minimum of 4 int
- Manual code does not check INF
  - Faster, simpler, INF check not needed
  - produces same answers
- 30% faster on RNA_STRAND 4666 molecules
- Genetic Improvement
  - GI 1.6% on top

Genetic Improvement of SSE code

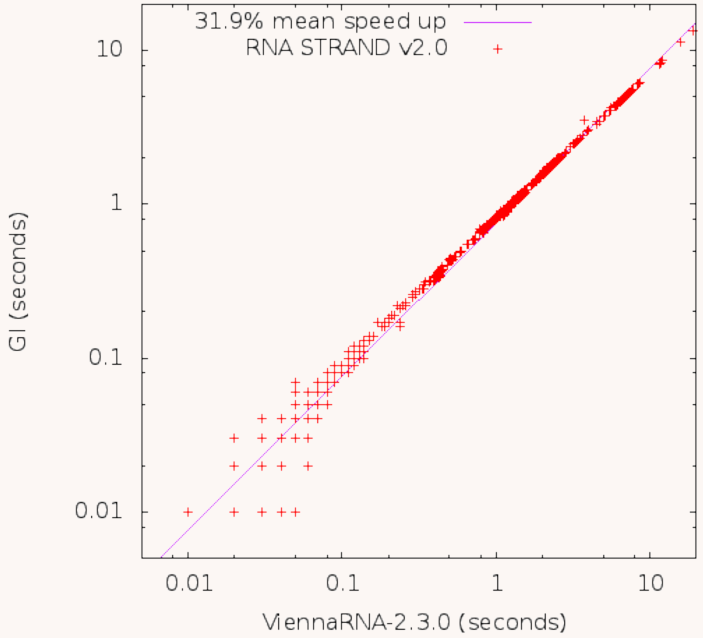
- Have 128 bit code wanted GI to generate 256/512 bit code but no hardware
- BNF grammar
  - – Intel library documentation: 789 rules
    - Evolution pulls in new code via its documentation.
    - Use function argument types and return value type to ensure evolved code is reasonable.
  - – Revision history of manual code: 168 rules
    - RCS steps from original to full SSE implementation
  - – Manual code: 141 rules
  - – Total 1030 rules

Fitness of mutant SSE code

- Run on input CRW_01455 through code to be improved run 4 232 596 times. Gives sequence of 4 232 596 fitness cases.
- Each generation
  - Split evenly into 5 subsequences
  - Choose a subsequence of 2000 tests at random from each → 10000 tests
- Test each mutant (15 second CPU limit)
  - 101 times on 10000 test cases. Check answers
  - 1st quartile elapsed time (nanosecond timer)

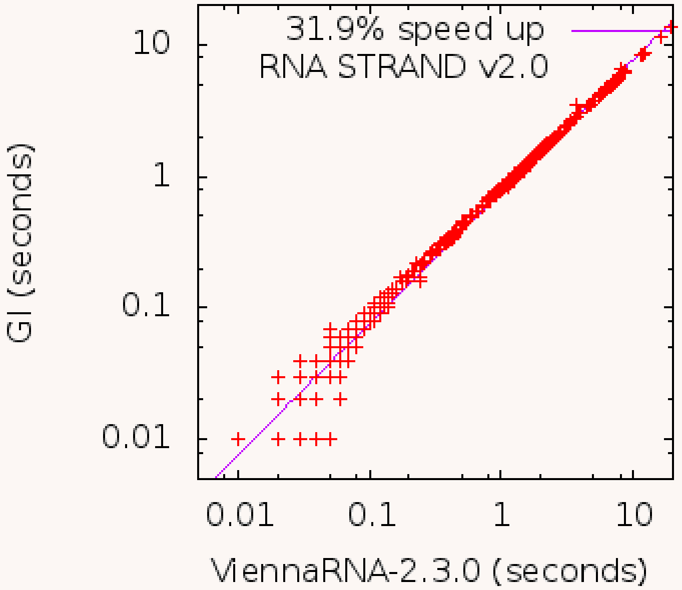
Evolved code

- Run overnight on otherwise idle ubuntu box.
- Tidy best in last population (gen 147)
- GI simplifies SSE code so executes exactly a multiple of 4 times.
- RNAfold gives same answer on 4666 molecules (GI code tested 138 billion times)



Summary

- 1st quartile gives robust average
- GGGP applied to state-of-the-art RNA tool.
- GI code base use revision history, library as well as manual code.
  - SSE on tiny fraction of whole code gives 30% speedup on real dataset.
  - GI modest improvement.
  - BUT only manual changes
- Expand to 512 bits?



END

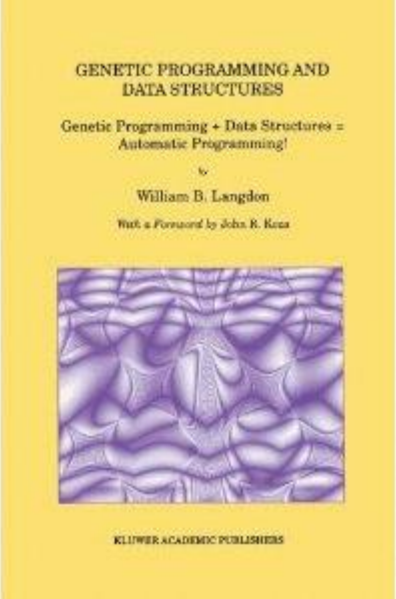
http://www.cs.ucl.ac.uk/staff/W.Langdon/ http://www.epsrc.ac.uk/ EPSRC

Genetic Improvement

W. B. Langdon

CREST

Department of Computer Science



GENETIC PROGRAMMING AND DATA STRUCTURES

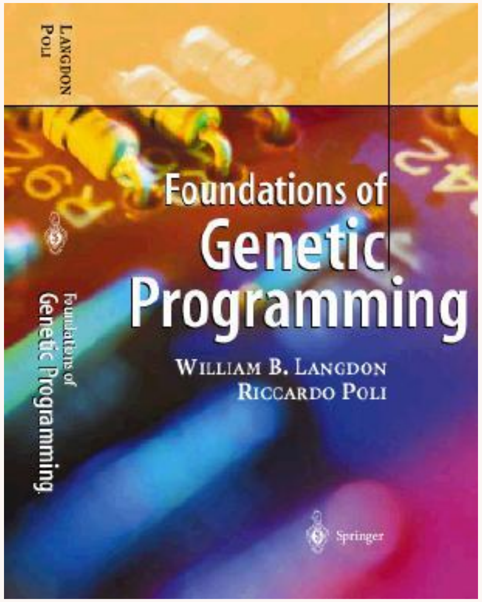
Genetic Programming + Data Structures = Automatic Programming!

by

William B. Langdon

With a Foreword by John R. Koza

KLUWER ACADEMIC PUBLISHERS

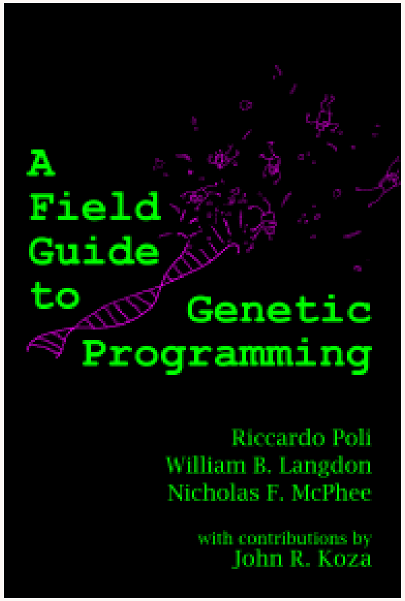


LANGDON
POLI

Foundations of Genetic Programming

WILLIAM B. LANGDON
RICCARDO POLI

Springer



A Field Guide to Genetic Programming

Riccardo Poli
William B. Langdon
Nicholas F. McPhee

with contributions by
John R. Koza

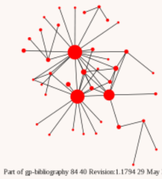
The Genetic Programming Bibliography

http://www.cs.bham.ac.uk/~wbl/biblio/

11628 references, 10000 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.

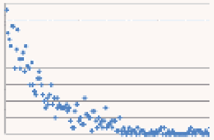


Co-authorships

Co-authorship community.
Downloads

Downloads by day

A personalised list of every author's GP publications.



Your papers

blog

Search the GP Bibliography at
http://liinwww.ira.uka.de/bibliography/Ai/genetic.programming.html