Genetic Improvement of GPU Software

W. B. Langdon
Computer Science, University College London

WIKIPEDIA
Genetic Improvement

GI 2017, Berlin,
15/16 July 2017
GECCO workshop

Based on GI special issue *forthcoming*

Volume 17, Number 4, December 2016 ISSN: 1389-2576

GENETIC PROGRAMMING AND EVOLVABLE MACHINES

Editor-in-Chief:
Lee Spector
Founding Editor:
Wolfgang Banzhaf

Springer

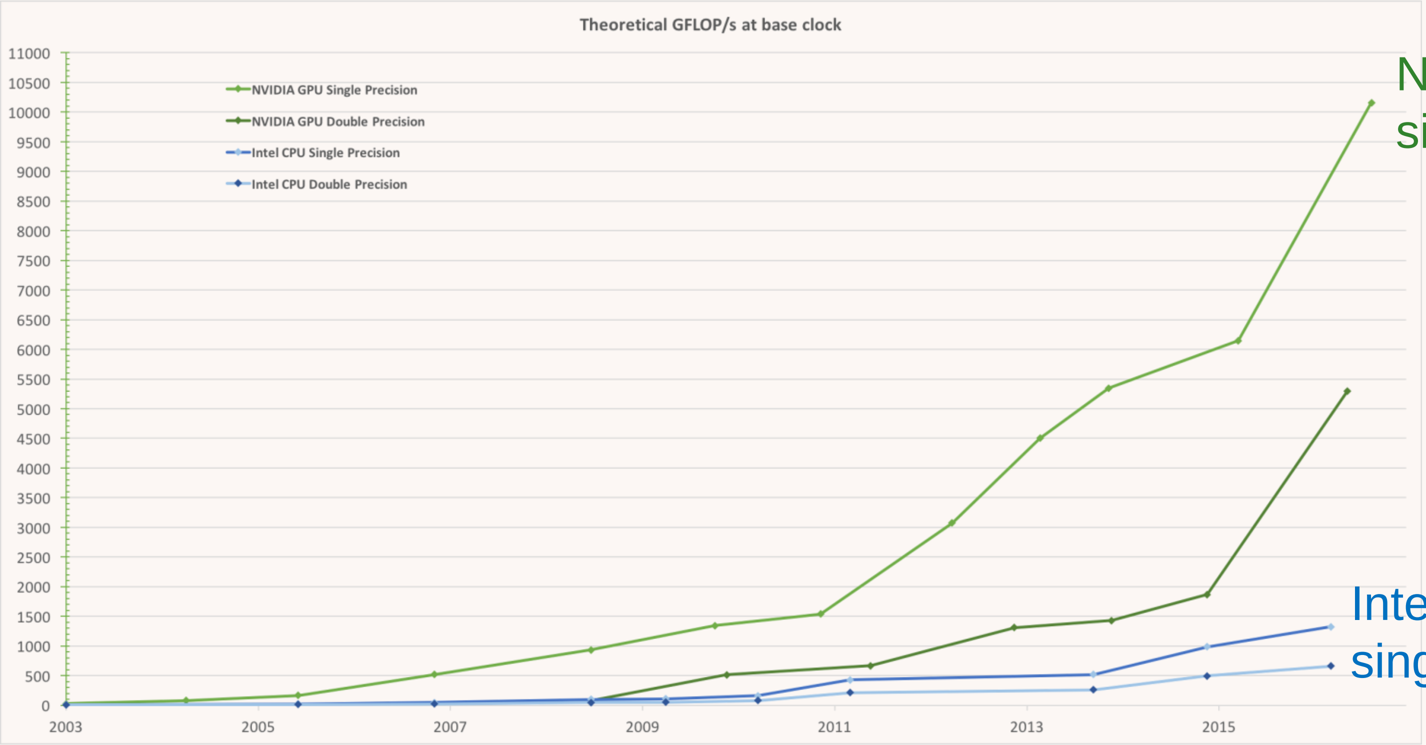
Genetic Improvement and GPGPU

- Why use graphics hardware? (speed)
- Difficulty of GPGPU programming

1. Automatically creating GPU code: gzip
2. Upgrade GPU software: StereoCamera
3. GI giving substantial improvement
   - 3D medical imaging, **BarraCUDA**
4. Grow and Graft Genetic Programming (GGGP) with human input
   - RNA folding x10000

Why use graphics hardware GPUs

Theoretical GFLOPS at base clock



Nvidia GPU single precision

Intel CPU single precision

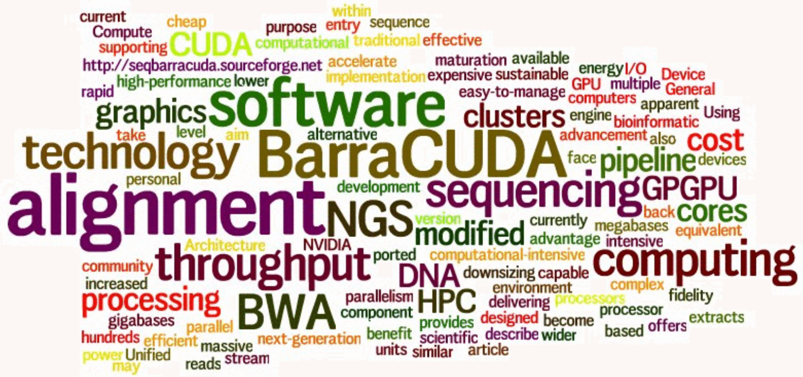
Floating-Point Operations per Second for the CPU and GPU

Nvidia CUDA 8.0 C Programming Guide

Performance GPGPU programming is hard

- High level (e.g. Matlab) speed from matrix algebra, matrix libraries.
- General purpose code CUDA (OpenCL)
- C like. Need to code many details.
- Hard to get right
- Hard to get performance
- Hard to keep performance, new hardware
  - Re-tune for next hardware generation

Genetically Improved BarraCUDA

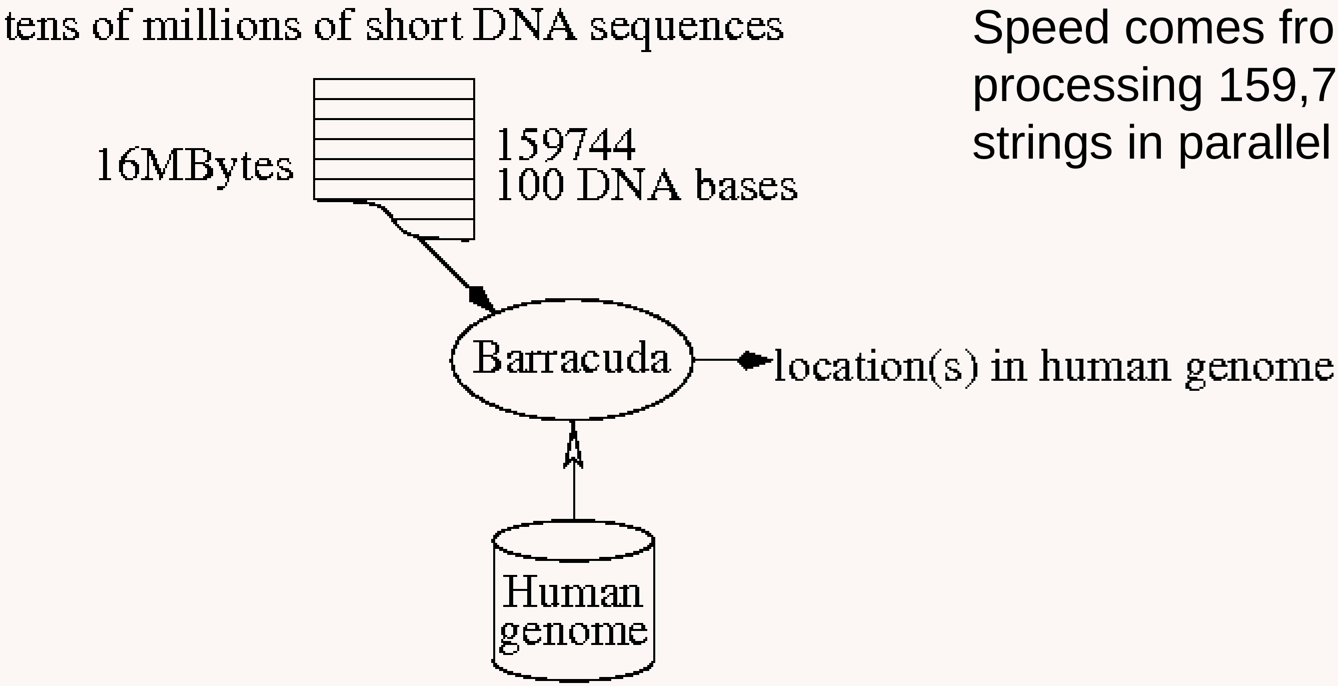


- Background
  - What is BarraCUDA
  - Using GI to improve parallel software, i.e. BarraCUDA
- Results
  - 100× speedup

What is BarraCUDA ?

DNA analysis program

- 8000 lines C code, SourceForge.
- Rewrite of BWA for nVidia CUDA

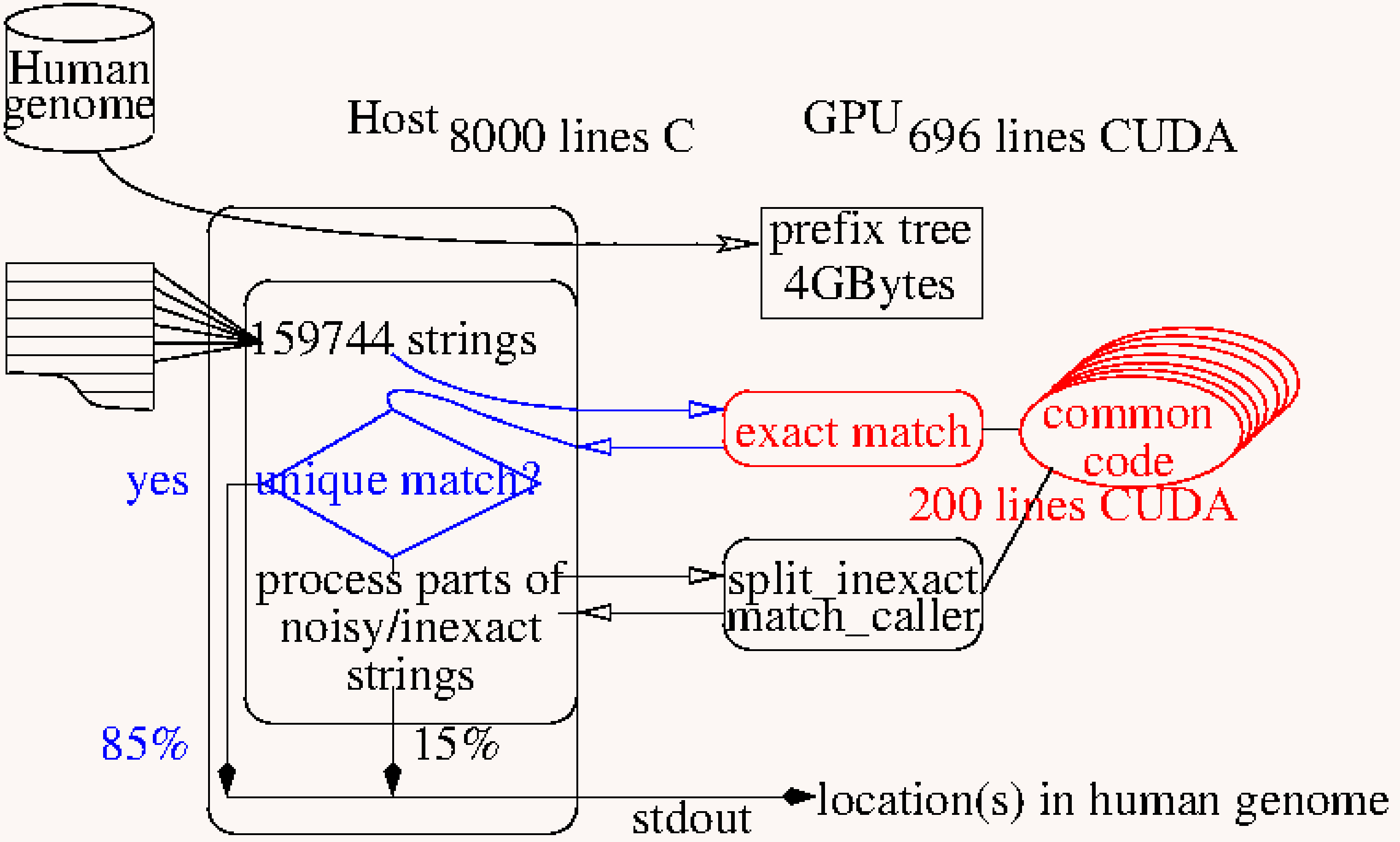

tens of millions of short DNA sequences
16MBytes
159744
100 DNA bases
Barracuda
location(s) in human genome
Human genome

Speed comes from processing 159,744 strings in parallel on GPU

BarraCUDA 0.7.107b

Manual host changes to call exact_match kernel

GI parameter and code changes on GPU



Why 1000 Genomes Project ?

- Data typical of modern large scale DNA mapping projects.
- Flagship bioinformatics project
  - Project mapped all human mutations.
- 604 billion short human DNA sequences.
- Download raw data via FTP

$120million 180Terra Bytes

Preparing for Evolution

- Re-enable exact matches code
- Support 15 options(conditional compilation)
- Genetic programming fitness testing framework
  - Generate and compile 1000 unique mutants
    - Whole population in one source file
    - Remove mutants who fail to compile and then re-run compiler to compile the others
  - Run and measure speed of 1000 kernels
    - Reset GPU following run time errors
  - For each kernel check 159444 answers

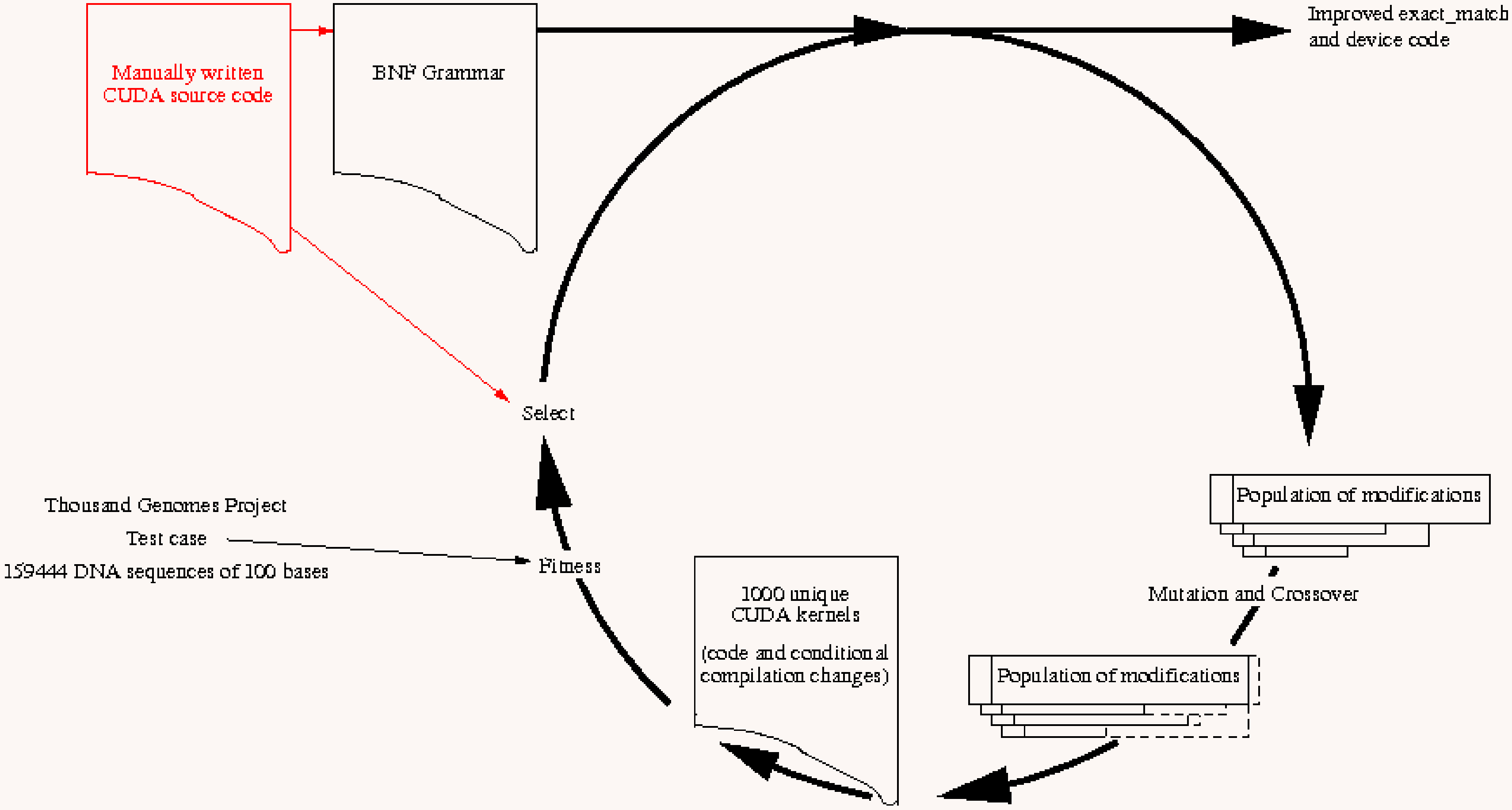
Fixed Parameters

Parameter		default	Lines of code affected
BLOCK_W	int	64	all
cache_threads	“” int	“”	44
kl_par	binary	off	19
occ_par	binary	off	76
many_blocks	binary	off	2
direct_sequence	binary	on	63
direct_index	binary	on	6
sequence_global	binary	on	16
sequence_shift81	binary	on	30
sequence_stride	binary	on	14
mycache4	binary	on	12
mycache2	binary	off	11
direct_global_bwt	binary	off	2
cache_global_bwt	binary	on	65
scache_global_bwt	binary	off	35

Evolving BarraCUDA kernel

- Convert manual CUDA code into grammar
- Grammar used to control code modification
- GP manipulates patches and fixed params
    - Small movement/deletion of existing code
    - New program source is syntactically correct
    - Automatic scoping rules ensure almost all mutants compile
    - Force loop termination
- Genetic Programming continues despite compilation and runtime errors

Evolving BarraCUDA



50 generations in 11 hours

BNF Grammar

Configuration parameter

```
if (*lastpos!=pos_shifted)
{
#ifndef sequence_global
  *data = tmp = tex1Dfetch(sequences_array, pos_shifted);
#else
  *data = tmp = Global_sequences(global_sequences,pos_shifted);
#endif /*sequence_global*/
  *lastpos=pos_shifted;
}
```

CUDA lines 119-127

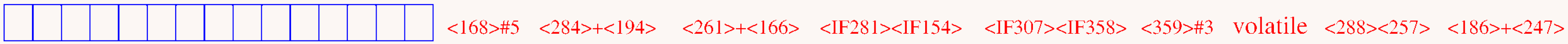
```
<119>   ::= " if" <IF_119> " \n"
<IF_119>::= "(*lastpos!=pos_shifted)"
<120>   ::= "{\n"
<121>   ::= "#ifndef sequence_global\n"
<122>   ::= "" <_122> "\n"
<_122>  ::= "*data = tmp = tex1Dfetch(sequences_array, pos_shifted);"
<123>   ::= "#else\n"
<124>   ::= "" <_124> "\n"
<_124>  ::= "*data = tmp = Global_sequences(global_sequences,pos_shifted);"
<125>   ::= "#endif\n"
<126>   ::= "" <_126> "\n"
<_126>  ::= "*lastpos=pos_shifted;"
<127>   ::= "}\n"
```

Fragment of Grammar (Total 773 rules)

9 Types of grammar rule

- Type indicated by rule name
- Replace rule only by another of same type
- 650 fixed, 115 variable.
- 43 statement (e.g. assignment, **Not** declaration)
- 24 IF
- <_392>   ::= " if" <IF_392> " {\n"
- <IF_392>   ::= " (par==0)"
- Seven for loops (for1, for2, for3)
- <_630>   ::= <okdeclaration_> <pragma_630> "for(" <for1_630> ";" "OK()&&" <for2_630> ";" <for3_630> ") \n"
- 2 ELSE
- 29 CUDA specials

Representation



- 15 fixed parameters; variable length list of grammar patches.
  - no size limit, so search space is infinite
- Uniform crossover and tree like 2pt crossover.
- Mutation flips one bit/int or adds one randomly chosen grammar change
- 3 possible grammar changes:
  - Delete line of source code (or replace by "", 0)
  - Replace with line of GPU code (same type)
  - Insert a copy of another line of kernel code

Example Mutating Grammar

```
<_947> ::= "*k0 = k;"
<_929> ::= "((int*)l0)[1] =
__shfl(((int*)&l)[1],threads_per_sequence/2,threads_per_sequence);
"
```

2 lines from grammar

```
<_947>+<_929>
```

Fragment of list of mutations

Says insert copy of line 929 before line 947

Copy of line 929

New code

```
((int*)l0)[1] =
__shfl(((int*)&l)[1],threads_per_sequence/2,threads_per_sequence);
*k0 = k;
```

Line 947

Summary

- Representation
  - 15 fixed genes (mix of Boolean and integer)
  - List of changes (delete, replace, insert). New rule must be of same type.
- Mutation
  - 1 bit flip or small/large change to int
  - append one random change to code
- Crossover
  - Uniform GA crossover
  - GP tree like 2pt crossover
- Evolve for 50 generations

Best K20 GPU Patch in gen 50

Parameter		new
scache_global_bwt	off	on
cache_threads	off	2
BLOCK_W	64	128

Store bwt cache in registers

Use 2 threads to load bwt cache

Double number of threads

line	Original Code	New Code
635		#pragma unroll
578	if(k == bwt_cuda.seq_len)	if(0)
947	`*k0 = k;`	`((int*)l0)[1] = __shfl(((int*)&l)[1],threads_per_sequence/2,threads_per_sequence);*k0 = k;`
126	`*lastpos=pos_shifted;`	

Line 578 `if` was never true

`l0` is overwritten later regardless

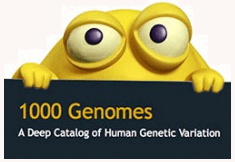
Change 126 disables small sequence cache 3% faster

Results

- Ten randomly chosen 100 base pair datasets from 1000 genomes project:
  - K20 1 840 000 DNA sequences/second (original 15000)
  - K40 2 330 000 DNA sequences/second (original 16 000)
- 100% identical
- manually incorporated into sourceForge

Conclusions

- On real typical data raw speed up > 100 times
  - Impact diluted by rest of code
  - On real data speed up to 3 times (arXiv.org)



- Incorporated into real system.1st GI in use.
  - 2753 sourceforge downloads (22 months).
  - Commercial use by Lab7 (in BioBuilds Nov2015)
  - IBM Power8
- Cambridge Epigenetix

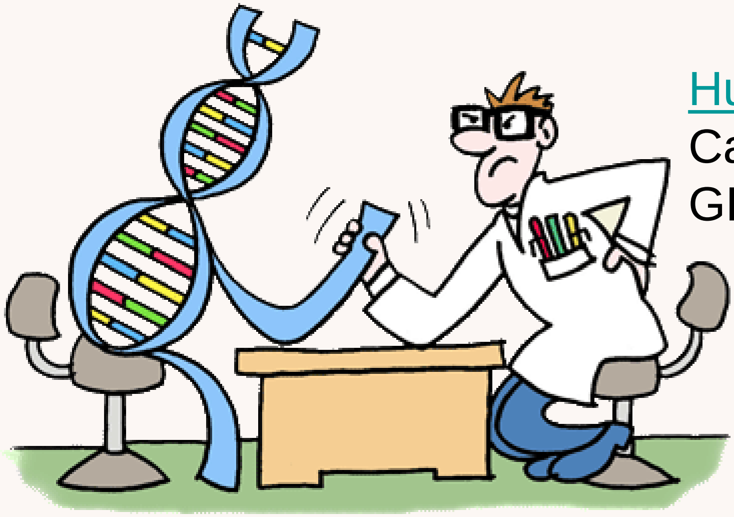
  - GTX 1080 21x faster than bwameth (twin core CPU)
  - Microsoft Azure GPU cloud

WIKIPEDIA
Genetic Improvement

Genetic Improvement 2017

GI 2017, Berlin,
15/16 July 2017
GECCO workshop

Submission due
29 March 2017



Humies: Human-Competitive
Cash prizes
GECCO-2017

END

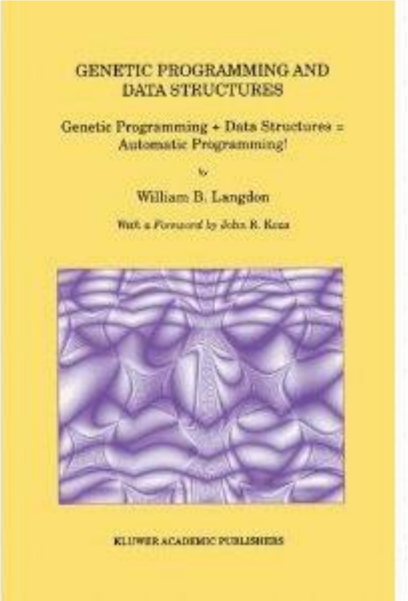
http://www.cs.ucl.ac.uk/staff/W.Langdon/ http://www.epsrc.ac.uk/

Genetic Improvement

W. B. Langdon

CREST

Department of Computer Science



GENETIC PROGRAMMING AND DATA STRUCTURES

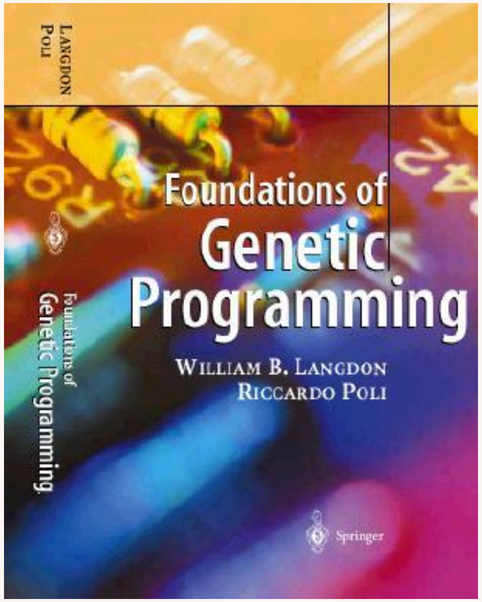
Genetic Programming + Data Structures = Automatic Programming!

by

William B. Langdon

With a Foreword by John R. Koza

KLUWER ACADEMIC PUBLISHERS



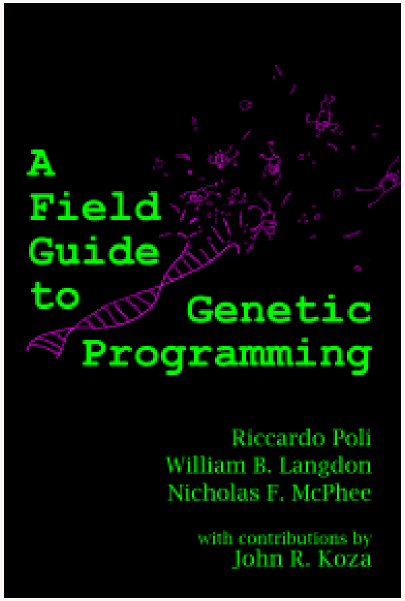
LANGDON POLI

Foundations of Genetic Programming

Foundations of Genetic Programming

WILLIAM B. LANGDON
RICCARDO POLI

Springer



A Field Guide to Genetic Programming

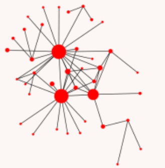
Riccardo Poli
William B. Langdon
Nicholas F. McPhee

with contributions by
John R. Koza

The Genetic Programming Bibliography

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

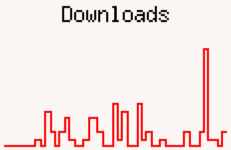
11315 references



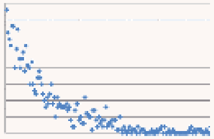
Part of gp-bibliography 84 40 Revision:1.1794 29 May 2011

RSS Support available through the Collection of CS Bibliographies. XML RSS

A web form for adding your entries. Co-authorship community. Downloads



A personalised list of every author’s GP publications.



blog

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