CREST

# Distilling GeneChips with GP on the Emerald GPU Supercomputer

W. B. Langdon

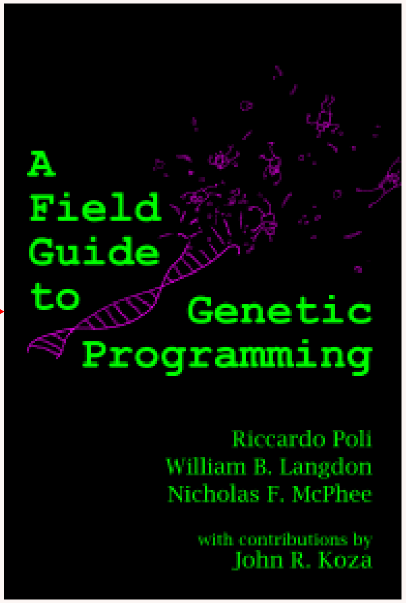
Centre for Research on Evolution, Search and Testing

Computer Science, UCL, London

8.2.2013

# Distilling GeneChips with GP on the Emerald GPU Supercomputer

- SIGEvolution 6(1) p15-21
- Not an introduction to genetic programming. Free pdf
- Run existing CUDA on Emerald
- 25× speedup


A Field Guide to Genetic Programming

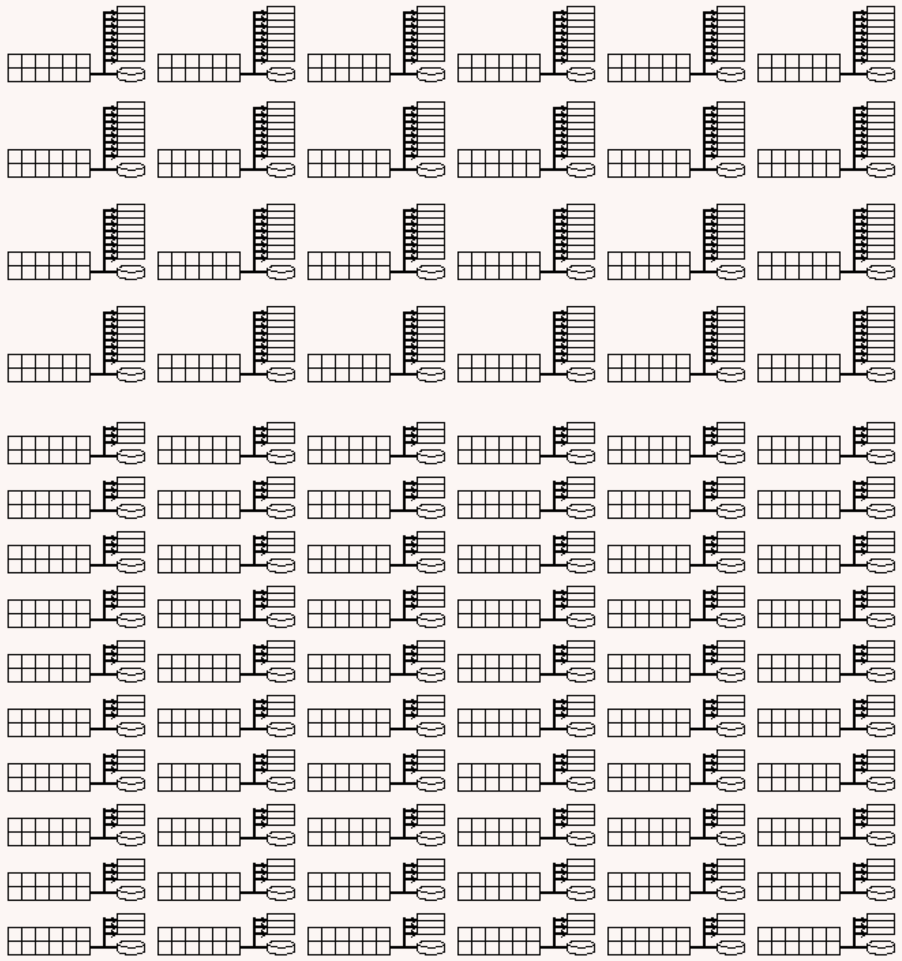
Riccardo Poli
William B. Langdon
Nicholas F. McPhee

with contributions by
John R. Koza

# Introduction

- 1987-89 all 251 women undergoing breast cancer surgery in Uppsala [ftp data].
- 1million data per woman. Predict who lives
- 2 passes winow useful data. Last builds final predictor [ftp code].
- Many independent runs in each pass. GPU used to process big population.

# Emerald



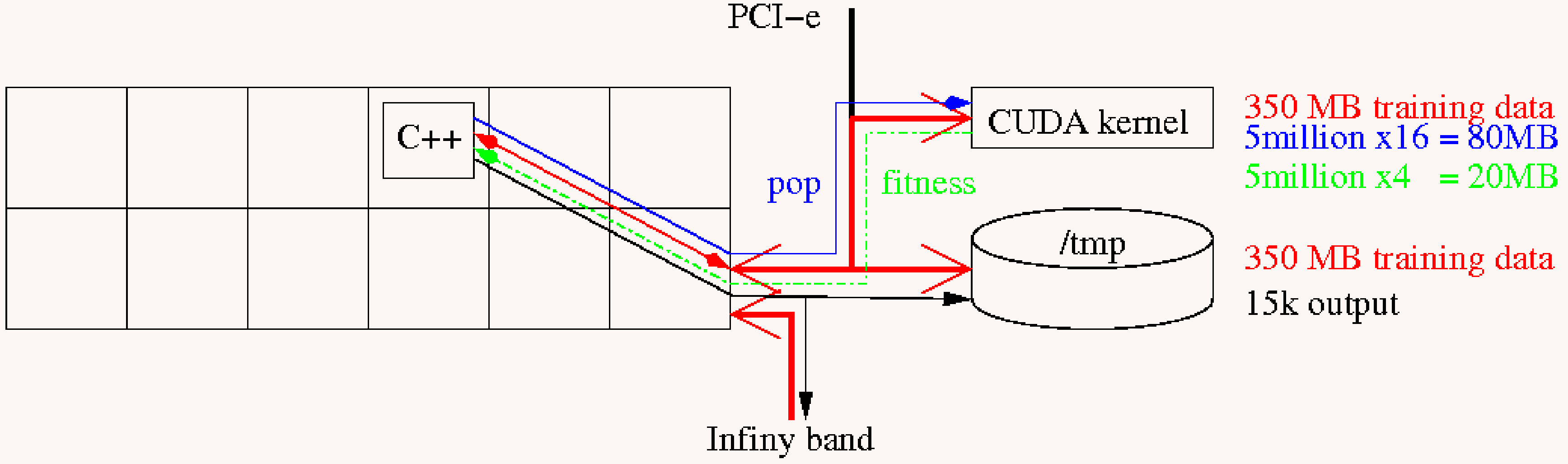
24

60

Each node consists of 6 twincore CPU (squares), local disk and 3 or 8 nVidia Tesla M2090 GPUs each containing 512 stream processors. 84 nodes (1008 CPU, 372 M2090, 190464 stream processors) connected by QDR Infiniband with Mellanox switches.

CREST

# GP run: one cpu↔gpu



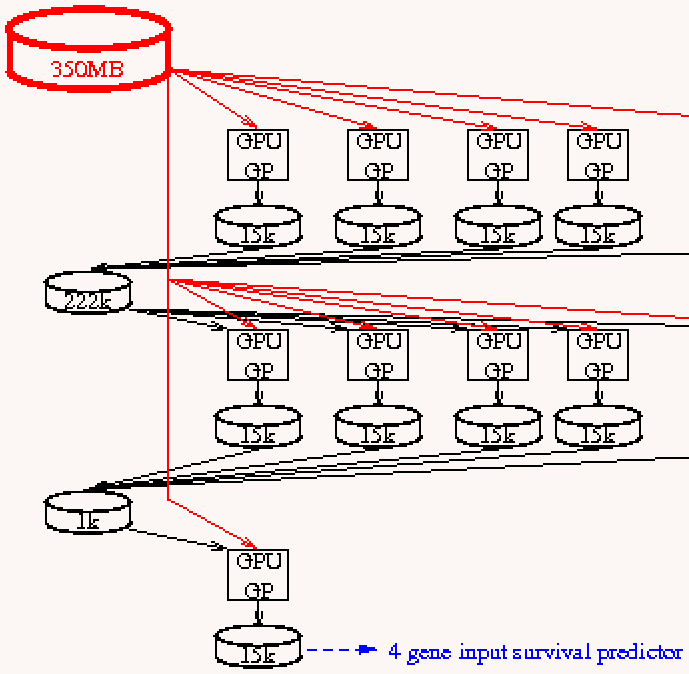
- Training data stored in GPU
- Each generation, new population created and transferred to GPU
- Each individual run on training data, gives fitness value.
- Each generation all fitness values transferred to host
- After 10 gen, best in pop reported (15k)

## GP run: one cpu↔gpu

- Training data stored in GPU
- Each generation, new population created and transfer to GPU
- Each individual run on training data, to give a fitness value.
- Each generation all fitness value transferred to host
- After 10 gen, best in pop reported

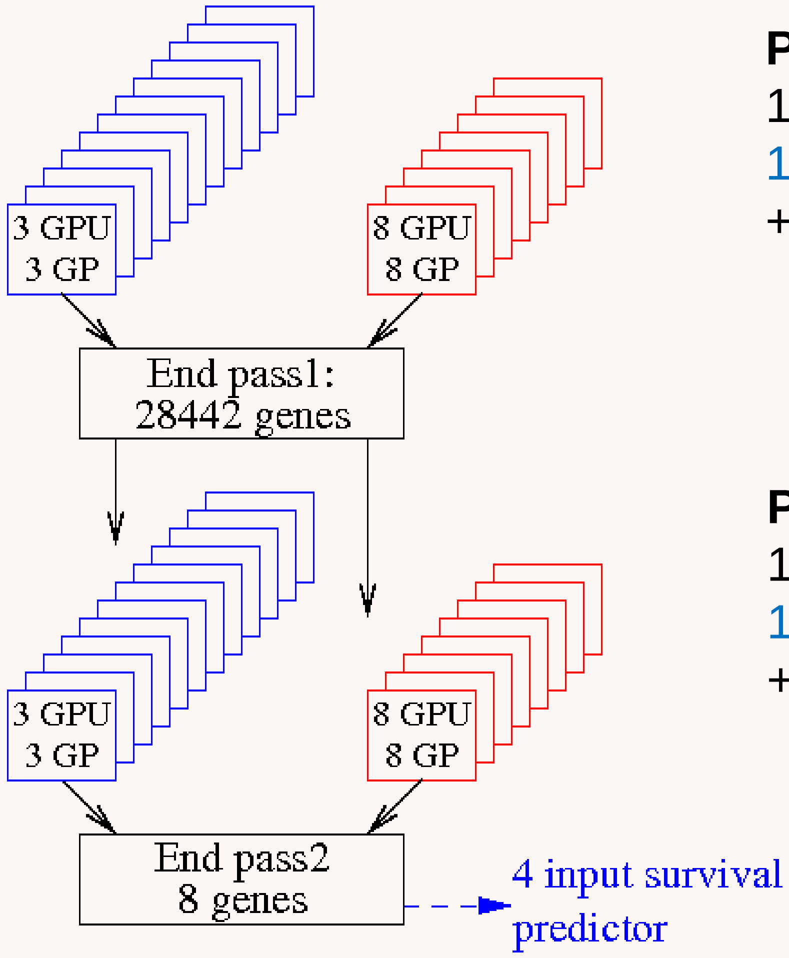
CREST

# Major data flows in datamining breast cancer



Phase 1 and 2 both have one hundred runs. (GeneChip training data in red.). The last phase (bottom) consists of a single GPU GP run which generates the final simple model which uses only four of the millions of GeneChip data to predict long term survival following breast tumour surgery.

# Mapping GP jobs to LSF and Emerald hardware

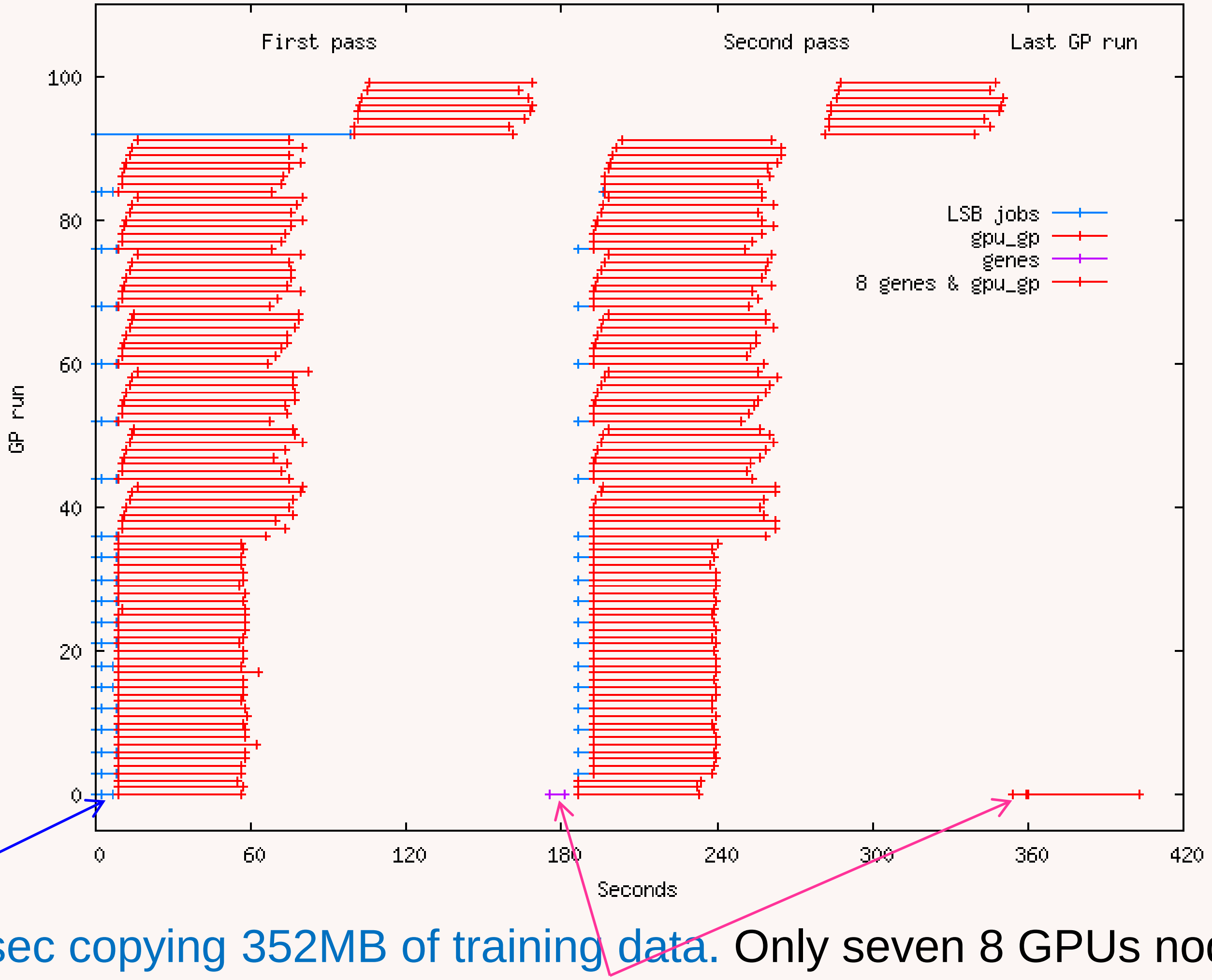


**Phase 1**
100 jobs = 12×3 + 8×8
12 nodes each has 3 GPU
+ 8 nodes each has 8 GPU

**Phase 2**
100 jobs = 12×3 + 8×8
12 nodes each has 3 GPU
+ 8 nodes each has 8 GPU

**Phase 3**
1 job

# 2×100+1 GP runs in parallel



First 1.6 sec copying 352MB of training data. Only seven 8 GPUs nodes were available, so GP runs 92–99 must wait. Gathering output of pass 1 and 2 takes 6 seconds. The final GP run finished 6 minutes 43 sec after the submitted to Emerald. Average 33.8 giga GPop/second.

# Recommendations

- Minimise copying datasets (cache on /tmp)
- Prevent core dumps  limit coredumpsize 0
- Emerald multi-user batch system:
  - not designed to give single user speedup
  - LSF batch queues shares CPUs not GPUs
- Use LSF to spread jobs across Emerald
- Use linux scripts to control multiple jobs on same node with fork “&” “wait”
- Make scripts resilient. Eg GPU already in use: so use a different one, retry.
- Other http://www.cs.ucl.ac.uk/staff/W.Langdon/emerald/

## Conclusions

- If application already consists of 100s of (semi) independent runs, easily transferred to Emerald (using LSF queues and unix scripts).
- 25× speedup without recoding for Emerald
- Workload needs to be big enough to warrant using Emerald.

# END

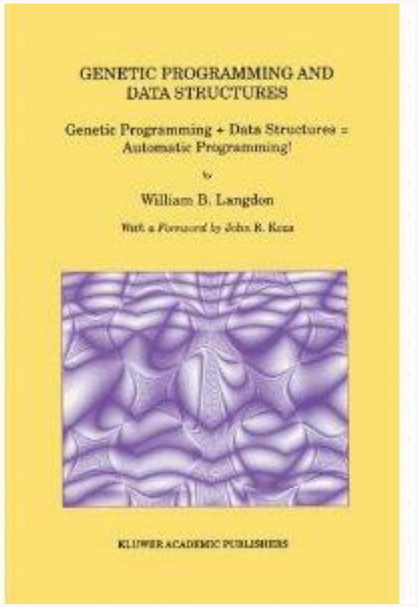
http://www.cs.ucl.ac.uk/staff/W.Langdon/ http://www.epsrc.ac.uk/

CREST

# Genetic Programming

UCL

W. B. Langdon
Department of Computer Science


GENETIC PROGRAMMING AND DATA STRUCTURES

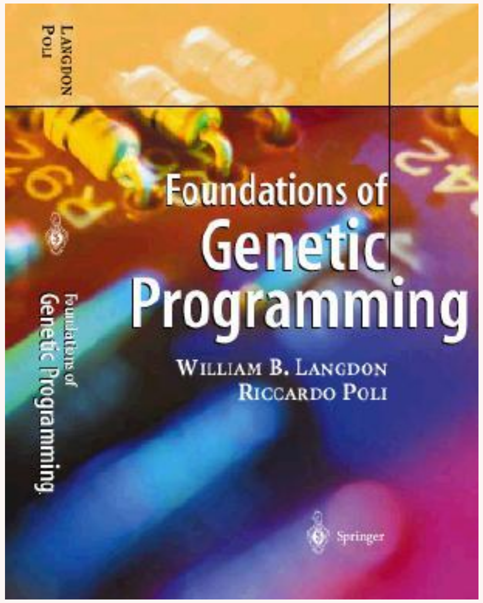
Genetic Programming + Data Structures = Automatic Programming!

by

William B. Langdon

With a Foreword by John R. Koza

KLUWER ACADEMIC PUBLISHERS

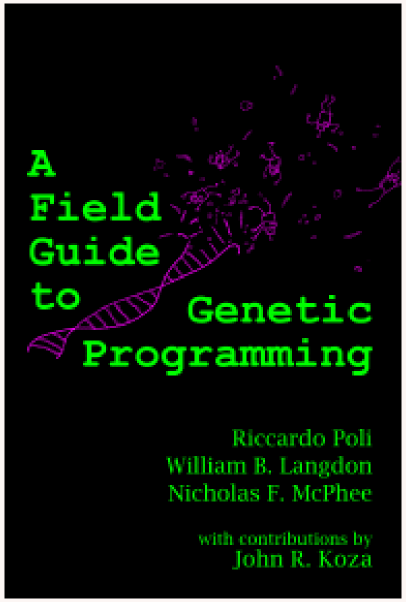

LANGDON POLI

Foundations of Genetic Programming

WILLIAM B. LANGDON
RICCARDO POLI

Foundations of Genetic Programming

Springer


A Field Guide to Genetic Programming

Riccardo Poli
William B. Langdon
Nicholas F. McPhee

with contributions by
John R. Koza