Hybrid CPU/GPU Acceleration of Detection of 2-SNP Epistatic Interactions in GWAS

Jorge González-Domínguez*, Bertil Schmidt*,
Jan C. Kässens**, Lars Wienbrandt**

*Parallel and Distributed Architectures Group, Johannes Gutenberg University of Mainz, Germany
{j.gonzalez,bertil.schmidt}@uni-mainz.de
**Department of Computer Science, Christian-Albrechts-University of Kiel, Germany
{jka,lwi}@informatik.uni-kiel.de

20th International Euro-Par Conference
Euro-Par 2014

JG|U

C|A|U

Genome-Wide Association Studies (I)

Analyses of genetic influence
on diseases

Genome-Wide Association Studies (I)

Analyses of genetic influence on diseases

- *M* individuals

Genome-Wide Association Studies (I)

Analyses of genetic influence on diseases

- M individuals
  - K cases

Genome-Wide Association Studies (I)

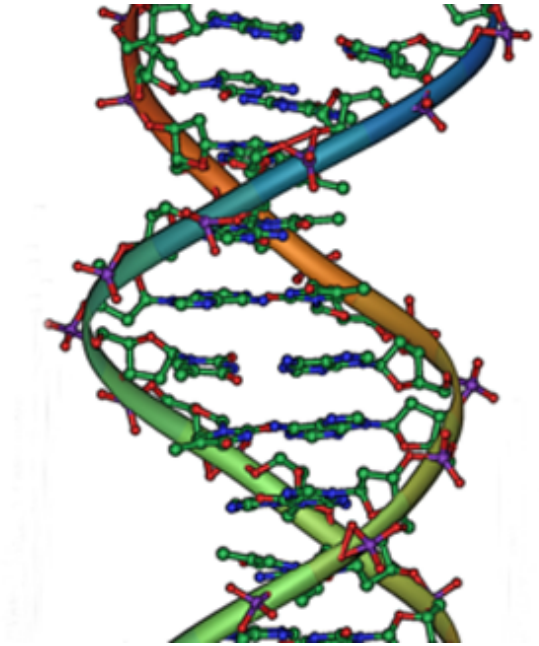
Analyses of genetic influence on diseases

- M individuals
  - K cases
  - C controls

Genome-Wide Association Studies (I)

Analyses of genetic influence on diseases

- *M* individuals
  - *K* cases
  - *C* controls
- *N* genetic markers, Single Nucleotide Polymorphisms (SNPs). 3 genotypes:
  - Homozygous Wild (w, AA, 0)
  - Heterozygous (h, Aa, 1)
  - Homozygous Variant (v, aa, 2)



Genome-Wide Association Studies (II)

	Cases						Controls									
SNP 1	0	1	2	0	1	2	0	1	2	0	1	2	0	1	2	1
SNP 2	0	1	1	0	2	0	0	0	1	2	2	1	0	1	1	2
SNP 3	0	0	0	0	0	0	0	0	1	2	1	1	1	2	1	1
SNP 4	0	1	0	1	0	1	0	1	2	2	2	2	1	1	1	1
SNP 5	0	2	2	2	0	1	1	1	1	0	0	1	1	0	2	2
SNP 6	1	0	1	0	1	0	1	0	1	2	1	2	1	2	2	1

Genome-Wide Association Studies (II)

	Cases						Controls									
SNP 1	0	1	2	0	1	2	0	1	2	0	1	2	0	1	2	1
SNP 2	0	1	1	0	2	0	0	0	1	2	2	1	0	1	1	2
SNP 3	0	0	0	0	0	0	0	0	1	2	1	1	1	2	1	1
SNP 4	0	1	0	1	0	1	0	1	2	2	2	2	1	1	1	1
SNP 5	0	2	2	2	0	1	1	1	1	0	0	1	1	0	2	2
SNP 6	1	0	1	0	1	0	1	0	1	2	1	2	1	2	2	1

Genome-Wide Association Studies (II)

	Cases						Controls									
SNP 1	0	1	2	0	1	2	0	1	2	0	1	2	0	1	2	1
SNP 2	0	1	1	0	2	0	0	0	1	2	2	1	0	1	1	2
SNP 3	0	0	0	0	0	0	0	0	1	2	1	1	1	2	1	1
SNP 4	0	1	0	1	0	1	0	1	2	2	2	2	1	1	1	1
SNP 5	0	2	2	2	0	1	1	1	1	0	0	1	1	0	2	2
SNP 6	1	0	1	0	1	0	1	0	1	2	1	2	1	2	2	1

Genome-Wide Association Studies (and III)

Definition

Two SNPs present epistasis or interaction if:

- Their joint genotype frequencies show a statistically significant difference between cases and controls which potentially explains the effect of the genetic variation leading to disease.
- The difference between cases and controls shown by the joint values is significantly higher than using only the individual SNP values.

BOOST

BOolean Operation-based Screening and Testing

- Binary traits
- Exhaustive search
- Statistical regression
- Good accuracy (used by biologists)
- Returns a list of SNP pairs with high interaction probability
- Fastest available tool. Intel Core i7 3.20GHz:
  - 40,000 SNPs and 3,200 individuals
    - About 800 million pairs
    - 51 minutes
  - 500,000 SNPs and 5,000 individuals
    - About 125 billion pairs (moderated size)
    - Estimated 12 days

GBOOST

CUDA version for GPUs

- Same accuracy as BOOST
- 40,000 SNPs and 6,400 individuals
  - About 800 million pairs
  - 28 seconds on a GTX Titan
- 500,000 SNPs and 5,000 individuals
  - About 125 billion pairs (moderated size)
  - 1 hour on a GTX Titan

GBOOST

CUDA version for GPUs

- Same accuracy as BOOST
- 40,000 SNPs and 6,400 individuals
  - About 800 million pairs
  - 28 seconds on a GTX Titan
- 500,000 SNPs and 5,000 individuals
  - About 125 billion pairs (moderated size)
  - 1 hour on a GTX Titan

High-throughput genotyping technologies collect few million SNPs of an individual within a few minutes $\rightarrow$ Expected datasets with 5M SNPs and 10,000 individuals

Goal of the Work

Development of EpistSearch, improving BOOST and GBOOST for GWAS

- Same accuracy
- CPU computation
  - Faster algorithm
  - Multithreaded version
- GPU computation
  - Faster algorithm
  - Improvement of the CUDA kernel
- CPU/GPU computation
  - Inter-task hybrid parallelism

Contingency Tables in (G)BOOST (I)

For each SNP-pair $\rightarrow$ Number of occurrences of each combination of genotypes

Cases	**SNP2=0**	**SNP2=1**	**SNP2=2**
SNP1=0	n_{000}	n_{010}	n_{020}
SNP1=1	n_{100}	n_{110}	n_{120}
SNP1=2	n_{200}	n_{210}	n_{220}
Controls	**SNP2=0**	**SNP2=1**	**SNP2=2**
SNP1=0	n_{001}	n_{011}	n_{021}
SNP1=1	n_{101}	n_{111}	n_{121}
SNP1=2	n_{201}	n_{211}	n_{221}

Contingency Tables in (G)BOOST (II)

SNP 4	0	1	0	1	0	1	0	1	2	2	2	2	1	1	1	1
SNP 6	1	0	1	0	1	0	1	0	1	2	1	2	1	2	2	1

Cases	SNP6=0	SNP6=1	SNP6=2
SNP4=0	0	4	0
SNP4=1	4	0	0
SNP4=2	0	0	0
Controls	**SNP6=0**	**SNP6=1**	**SNP6=2**
SNP4=0	0	0	0
SNP4=1	0	2	2
SNP4=2	0	1	2

Contingency Tables in (G)BOOST (III)

Boolean Representation of Genotype Data

- Applied in BOOST and GBOOST
- 6 strings per SNP
  - 3 per cases and 3 per controls (one per genotype {0,1,2})
  - One bit per individual
  - Represents whether the individual has the corresponding genotype

Contingency Tables in (G)BOOST (IV)

SNP 1	0	1	2	0	1	2	0	1	2	0	1	2	0	1	2	1

Contingency Tables in (G)BOOST (IV)

SNP 1	0	1	2	0	1	2	0	1	2	0	1	2	0	1	2	1

SNP 1 = 0;	1	0	0	1	0	0	1	0
SNP 1 = 1;	0	1	0	0	1	0	0	1
SNP 1 = 2;	0	0	1	0	0	1	0	0
SNP 1 = 0;	0	1	0	0	1	0	0	0
SNP 1 = 1;	0	0	1	0	0	1	0	1
SNP 1 = 2;	1	0	0	1	0	0	1	0

Contingency Tables in (G)BOOST (and V)

Drawback

- 50% memory overhead

Contingency Tables in (G)BOOST (and V)

Drawback

- 50% memory overhead

Advantage

- More efficient creation of contingency tables
- Only logical AND computations
  - Strings packed in arrays of 32 bits
  - Only $\frac{m}{32}$ 32-bit AND operations per value of the table
- n_{xy0} = (SNP 1=x) AND (SNP 2=y)

Contingency Tables in EpistSearch (I)

Optimization in EpisSearch

- Only 8 values of the contingency table explicitly calculated with AND
- Only four strings per SNP
- Additional information with the total count of each genotype for cases and controls (6 integers)
  - Calculated once per SNP when loading data
  - $sum_0, sum_1, sum_2, sum_0, sum_1, sum_2$

Contingency Tables in EpistSearch (II)

Cases	SNP2=0	SNP2=1	SNP2=2
SNP1=0	n_{000}	–	n_{020}
SNP1=1	–	–	–
SNP1=2	n_{200}	–	n_{220}
Controls	**SNP2=0**	**SNP2=1**	**SNP2=2**
SNP1=0	n_{001}	–	n_{021}
SNP1=1	–	–	–
SNP1=2	n_{201}	–	n_{221}

Contingency Tables in EpistSearch (II)

Cases	SNP2=0	SNP2=1	SNP2=2
SNP1=0	n_{000}	–	n_{020}
SNP1=1	–	–	–
SNP1=2	n_{200}	–	n_{220}
Controls	**SNP2=0**	**SNP2=1**	**SNP2=2**
SNP1=0	n_{001}	–	n_{021}
SNP1=1	–	–	–
SNP1=2	n_{201}	–	n_{221}

$$n_{010} = sum_0 - n_{000} - n_{020}$$

Contingency Tables in EpistSearch (and III)

Advantages

- Less memory requirements
- Faster calculation of the contingency tables
  - Only 8 values of the table need the $m/32$ 32-bit AND operations
  - The other values calculated with a few arithmetic operations

Filtering Stage in (G)BOOST (I)

Measuring interaction via log-linear models

Filtering Stage in (G)BOOST (I)

Measuring interaction via log-linear models

Log-Linear Measure (I)

$$\hat{L}_S - \hat{L}_H = N \sum_{ijk} \left[\hat{\pi}_{ijk} \log \left(\frac{\hat{\pi}_{ijk}}{\hat{p}_{ijk}} \right) \right]$$

- $\hat{L}_S$ log-likelihood of the saturated regression model
- $\hat{L}_H$ log-likelihood of the homogeneous association model
- $\hat{\pi}_{ijk}$ joint distribution obtained under the saturated model
- $\hat{p}_{ijk}$ distribution obtained under the homogeneous association model

Filtering Stage in (G)BOOST (II)

Measuring interaction via log-linear models

Log-Linear Measure (II)

$$\hat{L}_S - \hat{L}_H = N \sum_{ijk} \left[\hat{\pi}_{ijk} \log \left(\frac{\hat{\pi}_{ijk}}{\hat{p}_{ijk}} \right) \right]$$

- T the threshold for epistasis
- If $\hat{L}_S - \hat{L}_H > T \Rightarrow$ Epistasis

Filtering Stage in (G)BOOST (II)

Measuring interaction via log-linear models

Log-Linear Measure (II)

$$\hat{L}_S - \hat{L}_H = N \sum_{ijk} \left[\hat{\pi}_{ijk} \log \left(\frac{\hat{\pi}_{ijk}}{\hat{p}_{ijk}} \right) \right]$$

- T the threshold for epistasis
- If $\hat{L}_S - \hat{L}_H > T \Rightarrow$ Epistasis
- Computationally expensive
  - $\hat{p}_{ijk}$ computed through iterative methods

Filtering Stage in (G)BOOST (III)

Kirkwood Superposition Approximation (KSA)

- $\hat{L}_S - \hat{L}_{KSA} = N \sum_{ijk} \left[\hat{\pi}_{ijk} \log \left(\frac{\hat{\pi}_{ijk}}{\hat{p}^k_{ijk}} \right) \right]$
- $\hat{p}^k_{ijk} = \frac{1}{\eta} \frac{\pi_{ij.} \pi_{i.k} \pi_{.jk}}{\pi_{i..} \pi_{.j.} \pi_{..k}}$
- $\eta = \sum_{ijk} \frac{\pi_{ij.} \pi_{i.k} \pi_{.jk}}{\pi_{i..} \pi_{.j.} \pi_{..k}}$

Filtering Stage in (G)BOOST (III)

Kirkwood Superposition Approximation (KSA)

- $\hat{L}_S - \hat{L}_{KSA} = N \sum_{ijk} \left[\hat{\pi}_{ijk} \log \left(\frac{\hat{\pi}_{ijk}}{\hat{p}^k_{ijk}} \right) \right]$
- $\hat{p}^k_{ijk} = \frac{1}{\eta} \frac{\pi_{ij.} \pi_{i.k} \pi_{.jk}}{\pi_{i..} \pi_{.j.} \pi_{..k}}$
- $\eta = \sum_{ijk} \frac{\pi_{ij.} \pi_{i.k} \pi_{.jk}}{\pi_{i..} \pi_{.j.} \pi_{..k}}$
- Upper bound: $\hat{L}_S - \hat{L}_H \leq \hat{L}_S - \hat{L}_{KSA}$

Filtering Stage in (G)BOOST (III)

Kirkwood Superposition Approximation (KSA)

- $\hat{L}_S - \hat{L}_{KSA} = N \sum_{ijk} \left[\hat{\pi}_{ijk} \log \left(\frac{\hat{\pi}_{ijk}}{\hat{p}^k_{ijk}} \right) \right]$
- $\hat{p}^k_{ijk} = \frac{1}{\eta} \frac{\pi_{ij.} \pi_{i.k} \pi_{.jk}}{\pi_{i..} \pi_{.j.} \pi_{..k}}$
- $\eta = \sum_{ijk} \frac{\pi_{ij.} \pi_{i.k} \pi_{.jk}}{\pi_{i..} \pi_{.j.} \pi_{..k}}$
- Upper bound: $\hat{L}_S - \hat{L}_H \leq \hat{L}_S - \hat{L}_{KSA}$
- $\hat{L}_S - \hat{L}_{KSA} < T \Rightarrow$ No epistasis

Filtering Stage in (G)BOOST (III)

Kirkwood Superposition Approximation (KSA)

- $\hat{L}_S - \hat{L}_{KSA} = N \sum_{ijk} \left[\hat{\pi}_{ijk} \log \left(\frac{\hat{\pi}_{ijk}}{\hat{p}^k_{ijk}} \right) \right]$
- $\hat{p}^k_{ijk} = \frac{1}{\eta} \frac{\pi_{ij.} \pi_{i.k} \pi_{.jk}}{\pi_{i..} \pi_{.j.} \pi_{..k}}$
- $\eta = \sum_{ijk} \frac{\pi_{ij.} \pi_{i.k} \pi_{.jk}}{\pi_{i..} \pi_{.j.} \pi_{..k}}$
- Upper bound: $\hat{L}_S - \hat{L}_H \leq \hat{L}_S - \hat{L}_{KSA}$
- $\hat{L}_S - \hat{L}_{KSA} < T \Rightarrow$ No epistasis
- $\hat{L}_S - \hat{L}_{KSA}$ is computationally simpler and faster

Filtering Stage in (G)BOOST (and IV)

Pseudocode of (G)BOOST

Filtering Stage in (G)BOOST (and IV)

Pseudocode of (G)BOOST

- For each SNP-pair *P*
  1. Calculate Contingency Table of P

Filtering Stage in (G)BOOST (and IV)

Pseudocode of (G)BOOST

- For each SNP-pair P
  1. Calculate Contingency Table of P
  2. $v = KSA\_Value(P)$

Filtering Stage in (G)BOOST (and IV)

Pseudocode of (G)BOOST

- For each SNP-pair P
  1. Calculate Contingency Table of P
  2. $v = KSA\_Value(P)$
  3. If $v > T$
     1. $v = LogLinear\_Value(P)$

Filtering Stage in (G)BOOST (and IV)

Pseudocode of (G)BOOST

- For each SNP-pair P
  1. Calculate Contingency Table of P
  2. $v = KSA\_Value(P)$
  3. If $v > T$
     1. $v = LogLinear\_Value(P)$
     2. If $v > T$ include P in the output list as pair with epistasis

Filtering Stage in EpistSearch (I)

KSA's Superposition Approximation (KSASA)

- $D_{\mathrm{KL}}(E, O) = \sum_{ij} \pi_{ij1} \log\left(\frac{\pi_{ij1}}{\pi_{ij0}}\right)$
- E count of expected (control) studies
- O count of observed (case) studies
- D_{KL} is discrete Kullback-Leibler divergence

Filtering Stage in EpistSearch (I)

KSA's Superposition Approximation (KSASA)

- $D_{KL}(E, O) = \sum_{ij} \pi_{ij1} \log\left(\frac{\pi_{ij1}}{\pi_{ij0}}\right)$
- E count of expected (control) studies
- O count of observed (case) studies
- D_{KL} is discrete Kullback-Leibler divergence
- Upper bound: $\hat{L}_S - \hat{L}_H \leq \hat{L}_S - \hat{L}_{KSA} \leq N * D_{KL}(E, O)$

Filtering Stage in EpistSearch (I)

KSA's Superposition Approximation (KSASA)

- $D_{\mathrm{KL}}(E, O) = \sum_{ij} \pi_{ij1} \log\left(\frac{\pi_{ij1}}{\pi_{ij0}}\right)$
- E count of expected (control) studies
- O count of observed (case) studies
- D_{KL} is discrete Kullback-Leibler divergence
- Upper bound: $\hat{L}_{\mathrm{S}} - \hat{L}_{\mathrm{H}} \leq \hat{L}_{\mathrm{S}} - \hat{L}_{\mathrm{KSA}} \leq N * D_{\mathrm{KL}}(E, O)$
- Calculation of $N * D_{\mathrm{KL}}(E, O)$ even faster

Filtering Stage in EpistSearch (and II)

Pseudocode of EpistSearch

For each SNP-pair *P*

Filtering Stage in EpistSearch (and II)

Pseudocode of EpistSearch

For each SNP-pair *P*

1. Calculate Contingency Table of P

Filtering Stage in EpistSearch (and II)

Pseudocode of EpistSearch

For each SNP-pair P

1. Calculate Contingency Table of P
2. $v = KSASA\_Value(P)$

Filtering Stage in EpistSearch (and II)

Pseudocode of EpistSearch

For each SNP-pair P

1. Calculate Contingency Table of P
2. $v = KSASA\_Value(P)$
3. If $v > T$
   1. $v = KSA\_Value(P)$
   2. If $v > T$
      1. $v = LogLinear\_Value(P)$
      2. If $v > T$ include P in the output list as pair with epistasis

Parallel Implementations

- Each CPU/GPU core performs the whole calculation of different SNP-pairs
  - Calculation of the contingency table
  - Filtering

Parallel Implementations

- Each CPU/GPU core performs the whole calculation of different SNP-pairs
  - Calculation of the contingency table
  - Filtering
  - CPU multicore: PThreads
  - GPU: CUDA
  - CPU&GPU: CUDA&PThreads
    - GPU computes much more SNP-pairs than CPUs

CUDA Implementation (I)

CUDA kernel

- Genotyping information loaded in device memory through pinned copies
- In each kernel many SNP-pairs are analyzed
- Each thread performs the whole calculation of independent SNP-pairs

CUDA Implementation (I)

CUDA kernel

- Genotyping information loaded in device memory through pinned copies
- In each kernel many SNP-pairs are analyzed
- Each thread performs the whole calculation of independent SNP-pairs
- Only one kernel for all the computation

CUDA Implementation (I)

CUDA kernel

- Genotyping information loaded in device memory through pinned copies
- In each kernel many SNP-pairs are analyzed
- Each thread performs the whole calculation of independent SNP-pairs
- Only one kernel for all the computation
  - Thread divergence: only few threads need to compute the KSA and Log-Linear filters

CUDA Implementation (I)

CUDA kernel

- Genotyping information loaded in device memory through pinned copies
- In each kernel many SNP-pairs are analyzed
- Each thread performs the whole calculation of independent SNP-pairs
- Only one kernel for all the computation
  - Thread divergence: only few threads need to compute the KSA and Log-Linear filters
  - GBOOST solve it performing the Log-Linear filter on the CPUs
    - Contingency tables must be copied to host memory
    - Less performance

CUDA Implementation (II)

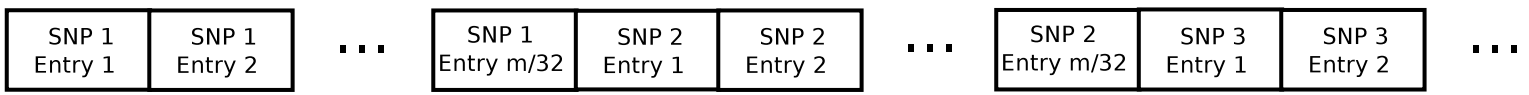
Format of the Genotyping Information

- 4 strings of binary values per SNP
  - 2 for controls and 2 for cases
  - 1 bit per individual
  - Represents whether the individual has the corresponding genotype ({0,2})

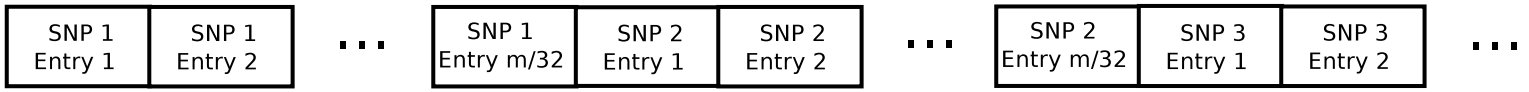
CUDA Implementation (II)

Format of the Genotyping Information

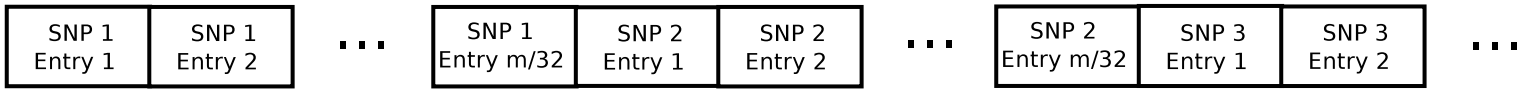
- 4 strings of binary values per SNP
  - 2 for controls and 2 for cases
  - 1 bit per individual
  - Represents whether the individual has the corresponding genotype ({0,2})
- For each string, information of 32 individuals packed in 32-bit arrays of length $m/32$



CUDA Implementation (and III)



CUDA Implementation (and III)



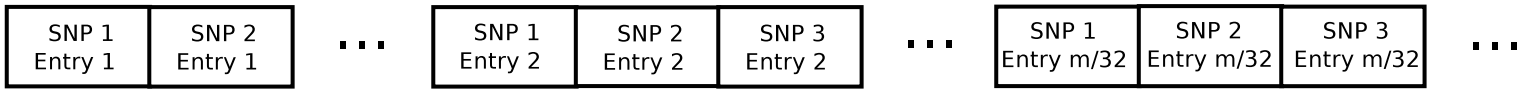
Increasing Coalescence

- Consecutive threads usually access to consecutive pairs
  - Stride of $m/32$
  - Bad coalescence

CUDA Implementation (and III)

Increasing Coalescence

- Consecutive threads usually access to consecutive pairs
  - Stride of $m/32$
  - Bad coalescence
- Entries of the arrays reordered when loading into device memory



System Characteristics

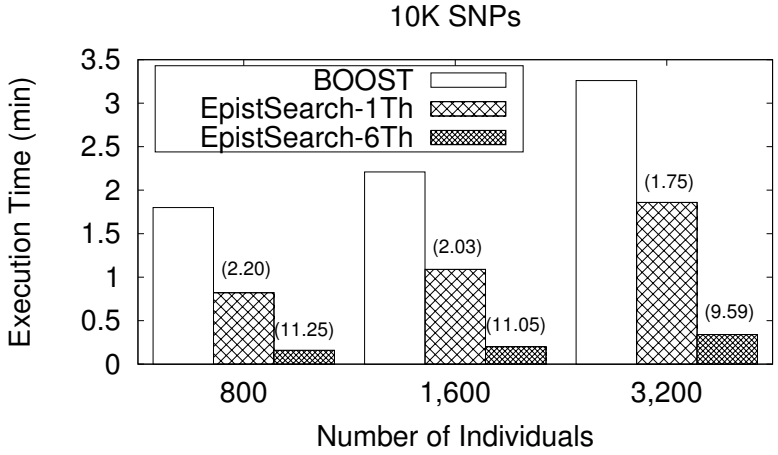
- Hex-core Intel Core i7 Sandy Bridge 3.20GHz
- 2 different NVIDIA GPUs (Kepler architecture):

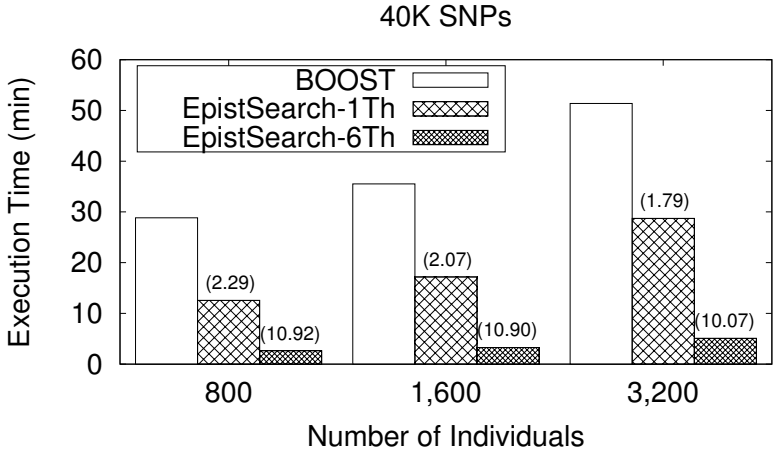
Name	Number of cores	Core frequency	Memory size
GTX 650Ti	768	980MHz	2GB
GTX Titan	2688	875.5MHz	6GB

Experiments for CPU

Table : Percentage of pairs that pass the KSASA and log-linear filters in the CPU experiments.

Num. Inds. →	800		1600		3200	
Num. SNPs →	**10K**	**40K**	**10K**	**40K**	**10K**	**40K**
KSASA	18.84	15.95	12.17	8.88	25.46	14.27
log-linear	11×10^{-4}	6×10^{-4}	27×10^{-4}	8×10^{-4}	170×10^{-4}	19×10^{-4}





Experiments for GPU (I)

Table : Percentage of pairs that pass the KSASA and log-linear filters in the GPU experiments.

Num. Inds. →	6400		12800		25600	
Num. SNPs →	**40K**	**160K**	**40K**	**160K**	**40K**	**160K**
KSASA	20.27	6.13	35.49	7.03	52.02	9.35
log-linear	110×10^{-4}	6×10^{-4}	800×10^{-4}	7×10^{-4}	4000×10^{-4}	12×10^{-4}

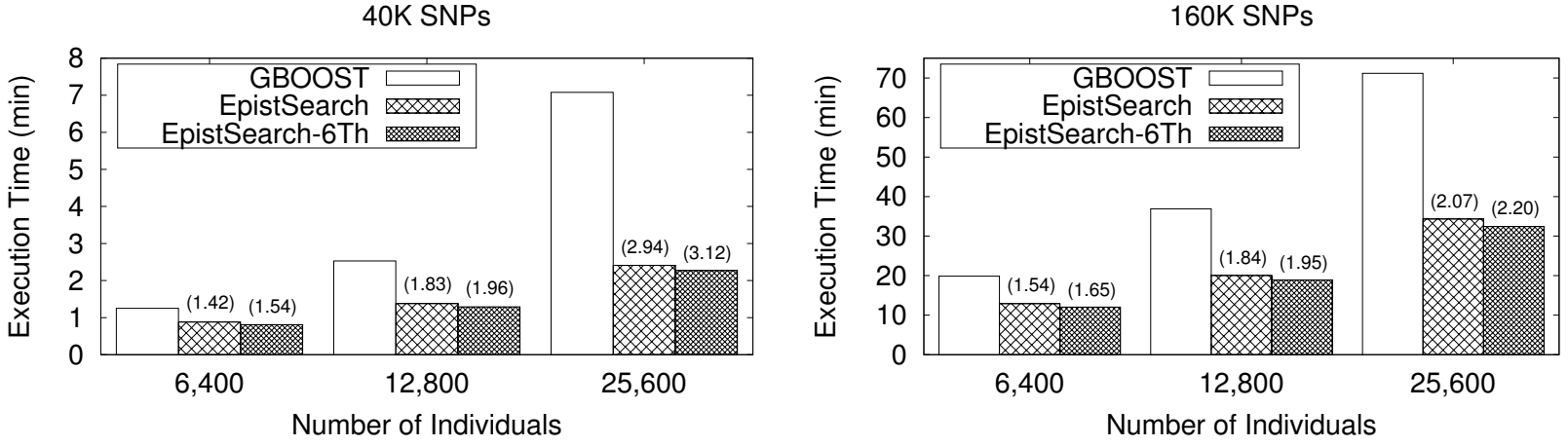


Figure : Execution times on the GTX 650 Ti GPU.

Experiments for GPU (II)

Table : Percentage of pairs that pass the KSASA and log-linear filters in the GPU experiments.

Num. Inds. →	6400		12800		25600	
Num. SNPs →	**40K**	**160K**	**40K**	**160K**	**40K**	**160K**
KSASA	20.27	6.13	35.49	7.03	52.02	9.35
log-linear	110×10^{-4}	6×10^{-4}	800×10^{-4}	7×10^{-4}	4000×10^{-4}	12×10^{-4}

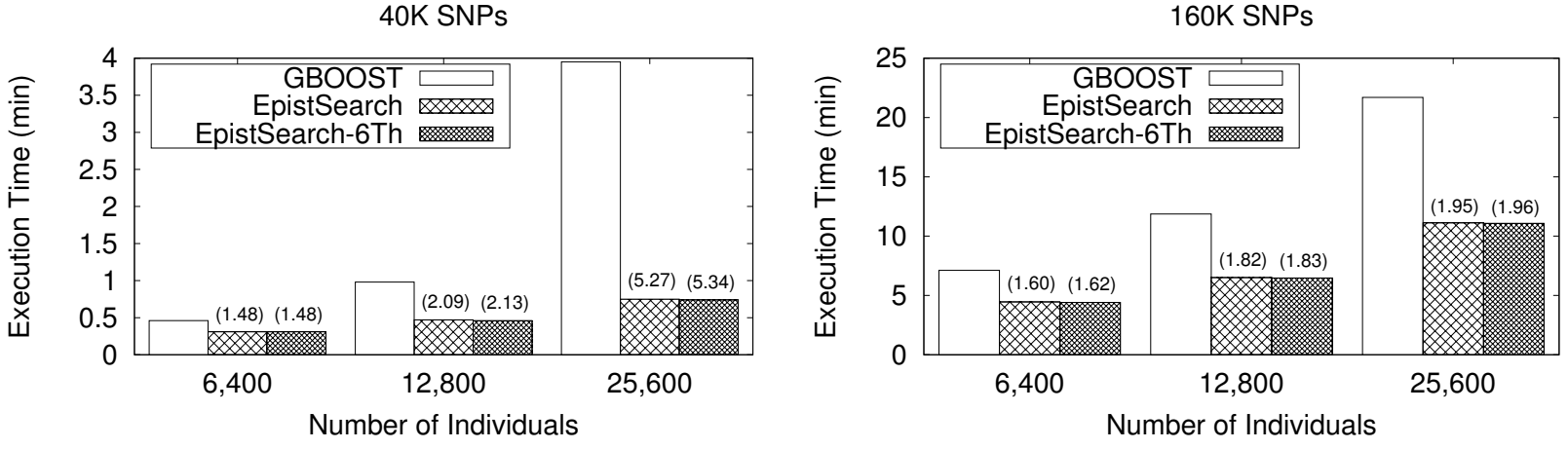


Figure : Execution times on the GTX Titan GPU.

Experiments for GPU (and III)

- Dataset with real information from the Wellcome Trust Case Control Consortium (WTCCC)
  - 500,568 SNPs
  - 2,005 cases with bipolar disorder
  - 3,004 controls

Tool	Architecture	Time	Speed (10^6 tests per second)
EpistSearch	GTX Titan + 6 Intel Core i7	42 m	49.81
EpistSearch	GTX Titan	43 m	49.04
GBOOST	GTX Titan	1 h 01 m	34.23
EpistSearch	GTX 650Ti + 6 Intel Core i7	1 h 48 m	19.29
EpistSearch	GTX 650Ti	1 h 57 m	17.81
GBOOST	GTX 650Ti	2 h 41 m	12.97
GBOOST*	GTX 285	2 h 43 m	12.81
EpiGPU*	GTX 580	2 h 55 m	11.90
SHEsisEPI*	GTX 285	27 h	1.29

Summary

- Development of EpistSearch
- Tool to search for epistasis between SNP-pairs in a fast manner taking advantage of CPU and GPU parallelism
  - Based on regression model
- EpistSearch improves (G)BOOST
  - Faster calculation of the contingency tables
  - Novel faster KSASA filter
  - Multithreaded CPU version
  - Log-linear filter also calculated on the GPU
  - Memory accesses more coalesced
  - Collaboration among CPU and GPU cores
- Able to reach very high speedups over (G)BOOST
  - 11.3 on CPU (with 6 cores)
  - 5.3 on a GTX Titan GPU
- Future work: multiGPU version

Hybrid CPU/GPU Acceleration of Detection of 2-SNP Epistatic Interactions in GWAS

Jorge González-Domínguez*, Bertil Schmidt*,
Jan C. Kässens**, Lars Wienbrandt**

*Parallel and Distributed Architectures Group, Johannes Gutenberg University of Mainz, Germany
{j.gonzalez,bertil.schmidt}@uni-mainz.de
**Department of Computer Science, Christian-Albrechts-University of Kiel, Germany
{jka,lwi}@informatik.uni-kiel.de

20th International Euro-Par Conference
Euro-Par 2014

JG|U

C|A|U