Fourth-Order Exhaustive Epistasis Detection for the xPU Era

Ricardo Nobre\textsuperscript{1}, Aleksandar Ilic\textsuperscript{1}, Sergio Santander-Jiménez\textsuperscript{2} and Leonel Sousa\textsuperscript{1}

\textsuperscript{1} INESC-ID, Instituto Superior Técnico, Universidade de Lisboa, Portugal
\{ricardo.nobre,aleksandar.ilic,leonel.sousa\}@inesc-id.pt

\textsuperscript{2} University of Extremadura (UNEX), Cáceres, Spain
sesaji@unex.es

50th International Conference on Parallel Processing (ICPP)
August 9-12, 2021 in Virtual Chicago, IL
Association Studies Identify Correlation w/ Genotype

<table>
<thead>
<tr>
<th>cases</th>
<th>controls</th>
</tr>
</thead>
<tbody>
<tr>
<td>have trait</td>
<td>do not have trait</td>
</tr>
</tbody>
</table>

markers (e.g. **SNPs**) 
most correlated w/ trait

output used in prevention / treatment, reducing spread of infectious diseases

**Single Nucleotide Polymorphism**

variation of single nucleotide in genome of $\geq 1\%$ of a population

<table>
<thead>
<tr>
<th>sample X</th>
<th>...</th>
<th>A</th>
<th>C</th>
<th>C</th>
<th>A</th>
<th>G</th>
<th>A</th>
<th>T</th>
<th>...</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>T</td>
<td>G</td>
<td>G</td>
<td>T</td>
<td>C</td>
<td>T</td>
<td>A</td>
<td></td>
</tr>
<tr>
<td>...</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>sample Y</th>
<th>...</th>
<th>G</th>
<th>T</th>
<th>A</th>
<th>G</th>
<th>T</th>
<th>A</th>
<th>G</th>
<th>...</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>C</td>
<td>A</td>
<td>T</td>
<td>C</td>
<td>A</td>
<td>T</td>
<td>C</td>
<td></td>
</tr>
</tbody>
</table>
Epistasis Detection as a Combinatorial Problem

Some phenotypes depend on high-order interactions\(^1,2\)

\[
\text{Total num comb. of SNPs:} \quad \frac{M!}{k! (M-k)!}
\]

2 alleles per SNP

- major allele (X) highest frequency
- minor allele (x) lowest frequency

3 Possible Genotypes

- Homozygous major: \(XX (0)\)
- Heterozygous: \(Xx (1)\)
- Homozygous minor: \(xx (2)\)

Embarrassingly parallel workload: repeat 3 steps

1. Count genotype freq.
2. Compute scores
3. Reduce scores

[2] C. Im, Genome-wide search for higher order epistasis as modifiers of treatment effects on bone mineral density in childhood cancer survivors. Eur J Hum Genet, 2018
Suitable for Heterogeneous Parallel Architectures

SoA for accelerators focuses on 2/3-way searches and is usually tied to proprietary technologies.

Proposed solution:
- Implement in SYCL for HW interop.
- Specialized to fourth-order explor. with support for GPU accelerators

CPU (Host) + GPU (Device) is a suitable pairing for achieving high epistasis detection throughput.
Contingency Table Construction (Example for 2-way)

Bitwise AND and POPC (count num. of set bits) instr. in CPUs\(^1\) and GPUs\(^2\)

POPC in SYCL through `sycl::popcount()`

Multiple samples processed per POPC instruction

\[3 \times (N_0 + N_1)\] bits per SNP to represent information for all cases and controls

Knowledge specific to input data can be used to reduce amount of instr. exec.

Remaining frequencies derived using simple arithmetic operations from:
- 16 genotypes (w/ SNPs of type 0 or 1)
- Precalculated freq. for third-order comb.

Close to $5 \times$ less inst. executed (from $\frac{81}{16}$) when processing challenging datasets

32 genotypes (4 x 8)

CALC_A_B_C(a, b, c, A_idx, B_idx, C_idx)
CALC_A_B_D(a, b, d, A_idx, B_idx, D_idx)
CALC_A_C_D(a, c, d, A_idx, C_idx, D_idx)
CALC_B_C_D(b, c, d, B_idx, C_idx, D_idx)

24 genotypes (6 x 4)

CALC_A_B_C(a, b, 2, A_idx, B_idx, C_idx)
CALC_A_B_C(a, 2, c, A_idx, B_idx, C_idx)
CALC_A_B_C(2, b, c, A_idx, B_idx, C_idx)
CALC_A_B_D(a, 2, d, A_idx, B_idx, D_idx)
CALC_A_B_D(2, b, d, A_idx, B_idx, D_idx)
CALC_A_C_D(2, c, d, A_idx, C_idx, D_idx)

8 genotypes (4 x 2)

CALC_A_B_C(a, 2, 2, A_idx, B_idx, C_idx)
CALC_A_B_C(2, b, 2, A_idx, B_idx, C_idx)
CALC_A_B_C(2, 2, c, A_idx, B_idx, C_idx)
CALC_B_C_D(2, 2, d, B_idx, C_idx, D_idx)

1 genotype (1 x 1)

CALC_A_B_C(2, 2, 2, A_idx, B_idx, C_idx)
# Experimental Setup and Performance Metric

<table>
<thead>
<tr>
<th>Systems</th>
<th>Intel CPU (arch.)</th>
<th>iGPU / dGPU (arch.)</th>
<th>DRAM</th>
<th>Operating System</th>
</tr>
</thead>
<tbody>
<tr>
<td>S1</td>
<td>(2×) Xeon Gold 6128 (Skylake) 2×6</td>
<td>3.4–3.7GHz</td>
<td></td>
<td>192GB DDR4</td>
</tr>
<tr>
<td>S2</td>
<td>Xeon E-2176G (Coffee Lake) 6</td>
<td>3.7–4.7GHz</td>
<td>Intel UHD Graphics P630 (Gen9.5) 192 (1200MHz)</td>
<td>system shared</td>
</tr>
<tr>
<td>S3</td>
<td>Core i9-10920X (Cascade Lake) 12</td>
<td>3.5–4.6GHz</td>
<td>Intel Iris Xe MAX (Gen12) 768 (1650MHz)</td>
<td>4GB (68.26 GB/s)</td>
</tr>
<tr>
<td>S4</td>
<td>Core i9-10980XE (Cascade Lake) 18</td>
<td>3.0–4.6GHz</td>
<td>NVIDIA Titan RTX (Turing) 4608 (1770MHz)</td>
<td>24GB (672.0 GB/s)</td>
</tr>
</tbody>
</table>

SYCL source code compiled with the Intel OneAPI DPC++ compiler

Runs performed with synthetic datasets with half cases/controls

Performance metric is the number of combinations of SNPs processed per second (scaled to sample size)
Experimental Results w/ GPU Acceleration

Best param. depends on the targeted GPU:
• 16-bit improved perf. on Titan RTX
• Perf. drop on UHD P630 / Iris Xe MAX

Similar perf. per stream core on Gen12 and Turing GPU architectures

Up to 2245 giga ($\times 10^9$) sets of SNPs per second on system S4 (Titan RTX)
Performance Using CPU as the SYCL Device

Overall higher perf. than BitEpi¹

1.33 × (Xeon Gold) and 1.13 × (Core i9)

BitEpi uses a data repres. and calculation method specialized for fast execution on modern CPUs

Higher performance on CPUs while achieving efficient execution on GPU devices

Conclusions

Higher perf. on CPU than related art

(1.33 × faster on Xeon Gold server)

Higher perf. on CPU than related art

(65 × speedup w/ GPU)

Use of GPUs provides significant perf. increase

Ongoing work

- reduce memory requirements of the proposed algorithm
- add support for heterogeneous clusters w/ additional architectures (e.g. custom HW on FPGAs)
Thank you!

This work was supported by the FCT (Fundação para a Ciência e a Tecnologia, Portugal), ERDF (European Regional Development Fund, EU) and EuroHPC Joint Undertaking through the projects UIDB/50021/2020, LISBOA-01-0145-FEDER-031901 (PTDC/CCI-COM/31901/2017, HiPeRBio) and grant agreement No 956213 (SparCity). We would also like to thank Intel Corporation for DevCloud and Iris Xe MAX GPU early access.

Related work by the same authors:

