IEEE World Congress on Computational Intelligence 2016

100 Million Dimensions Large-Scale Global Optimization Using Distributed GPU Computing

Alberto Cano and Carlos García-Martínez

Alberto Cano
Assistant Professor
Department of Computer Science
Virginia Commonwealth University, USA
acano@vcu.edu

IEEE WCCI 2016
Vancouver Canada

- Motivation
  - IEEE CEC Competition on Large Scale Global Optimization
    - 2013: 15 benchmark functions on 1000 dimensions
    - Imbalanced subcomponents, nonuniform sizes, conforming and conflicting overlapping functions
  - Lastra et al. proposed a GPU version of the MA-SW-Chains
    - Increase the dimensionality to up to 3 million dimensions
    - Limited to one GPU's memory and compute capability
    - Limited to functions with no subcomponents

M. Lastra, D. Molina, and J. M. Benítez, A high performance memetic algorithm for extremely high-dimensional problems, Information Sciences, vol. 293, pp. 35–58, 2015.

- Research questions
  - Why? Why not? Exploring the boundaries of LSGO
  - What are the dimensionalities limitations?
    - Memory:
      - Individual representation: O(D) using FP32/FP64
      - Population and offspring
      - GPU limited memory ≤ 12 GB
    - Computation:
      - Single fitness evaluation time
      - Running 3 million evaluations
      - GPU cores count ≤ 3k
  - Heuristics for 1,000D would work in a million-dimension problem?

- Benefits of LSGO parallelization
  - Speeding up existing algorithms
    - Speed up computation, reduce execution time
    - Same fitness results, “waste” of GPU resources for low dims
  - Generating multiple solutions per iteration
    - Multiple crossover, offspring, parallel local search
    - Occupancy maximization of GPU resources
    - Should lead to better fitness results
    - Cannot be fairly compared for equal number of FEs
  - Scaling to larger dimensionalities
    - Explore heuristics and convergence in high-dimensional sets
    - Occupancy maximization of GPU resources

- Proposal
  - Distributed GPU computing for scaling MA-SW-Chains to up to 100 million variables while maximizing GPU occupancy
  - SSGA to Generational: P individuals selected, crossed, mutated
  - Local Search: P individuals are optimized (in parallel)
  - Scaling definition of functions to millions of dimensions

- Speeding up fitness computation
  - Requisites: known expression and additively decomposable
  - Can be expressed as the sum of other functions
  - Sum of subcomponents computed in parallel

- Fully-separable functions

f_1: **Shifted Elliptic Function**

$$f_1(\mathbf{z}) = \sum_{i=1}^{D} 10^{6\frac{i-1}{D-1}} z_i^2$$

$$\sum \begin{cases} f_{X_1 \ldots X_j} \\ f_{X_j \ldots X_k} \\ f_{X_k \ldots X_D} \end{cases}$$

$$\mathbf{z} = T_{\text{osz}}(\mathbf{x} - \mathbf{x}^{\text{opt}})$$

$$\mathbf{x} \in [-100, 100]^D$$

- Definition of the function scales to any dimensionality with any number of parallel tasks

- Partially additively separable / overlapping functions

f_4: **7-nonseparable, 1-separable Shifted and Rotated Elliptic Function**

$$\mathcal{S} = \{50, 25, 25, 100, 50, 25, 25, 700\}$$

$$f_4(\mathbf{z}) = \sum_{i=1}^{|\mathcal{S}|-1} w_i f_{\text{elliptic}}(\mathbf{z}_i) + f_{\text{elliptic}}(\mathbf{z}_{|\mathcal{S}|})$$

$$\mathbf{y} = \mathbf{x} - \mathbf{x}^{\text{opt}}$$

$$\mathbf{y}_i = \mathbf{y}(\mathcal{P}_{[\mathcal{C}_{i-1}+1]} : \mathcal{P}_{[\mathcal{C}_i]}),\ i \in \{1, \ldots, |\mathcal{S}|\}$$

$$\mathbf{z_i} = T_{\text{osz}}(\mathbf{R}_i \mathbf{y_i}),\ i \in \{1, \ldots, |\mathcal{S}| - 1\}$$

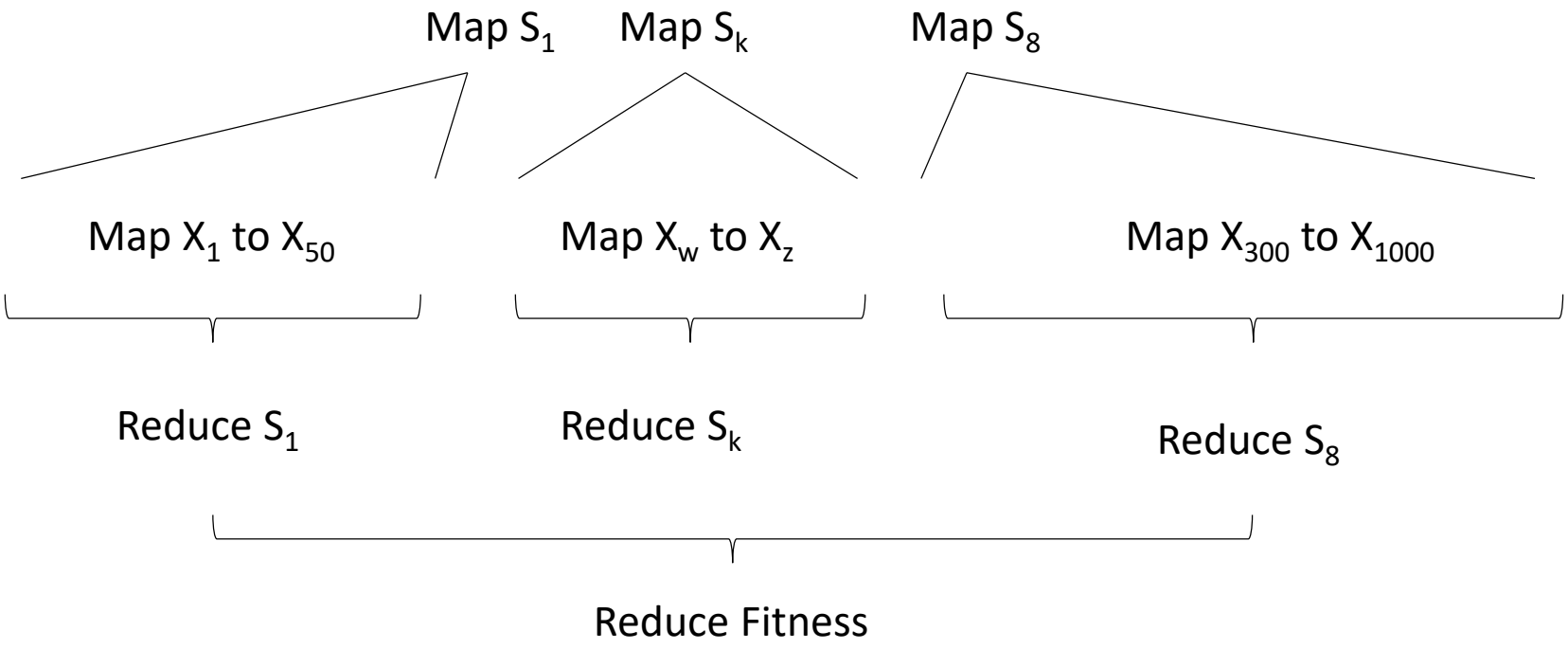
$$\mathbf{z}_{|\mathcal{S}|} = T_{\text{osz}}(\mathbf{y}_{|\mathcal{S}|})$$

$\mathbf{R}_i$: a $|\mathcal{S}_i| \times |\mathcal{S}_i|$ rotation matrix

- Thread computes X_i to Z_i
- Parallel Rotation Matrix
- Scaling using repetitions of the 1,000D block definition

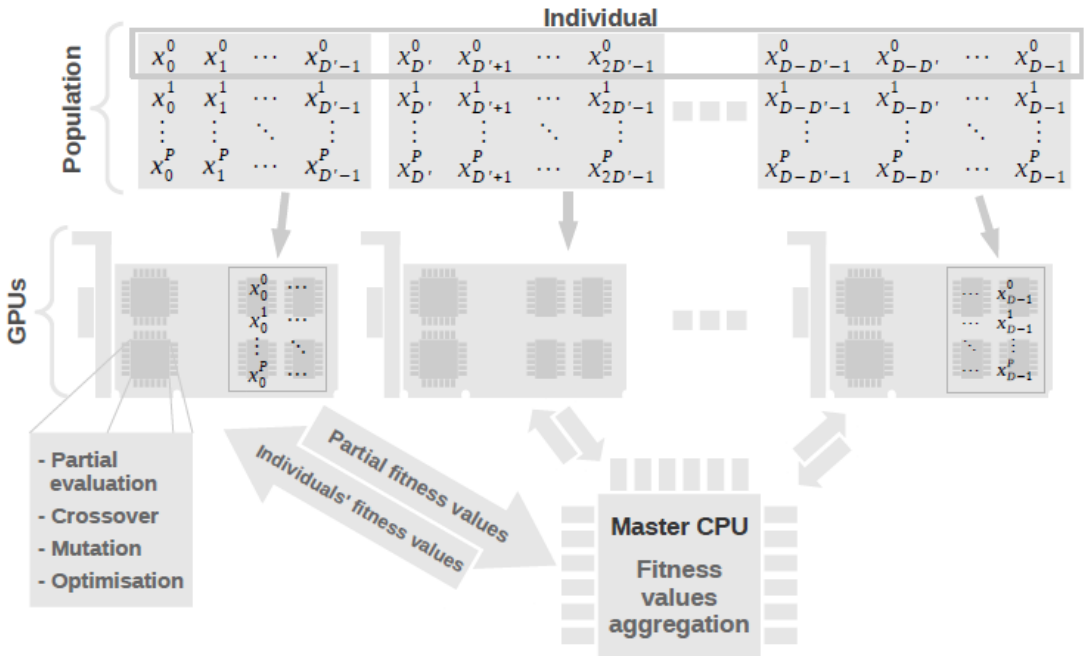
- Single-GPU parallelization of the fitness computation

$$\mathcal{S} = \{50, 25, 25, 100, 50, 25, 25, 700\}$$



- Concurrent kernels and asynchronous data / execution, no divergence
- Individuals and dimensions are mapped into a 2D grid of thread blocks
- Similar mapping for crossover and mutation: 1 thread – 1 dimension

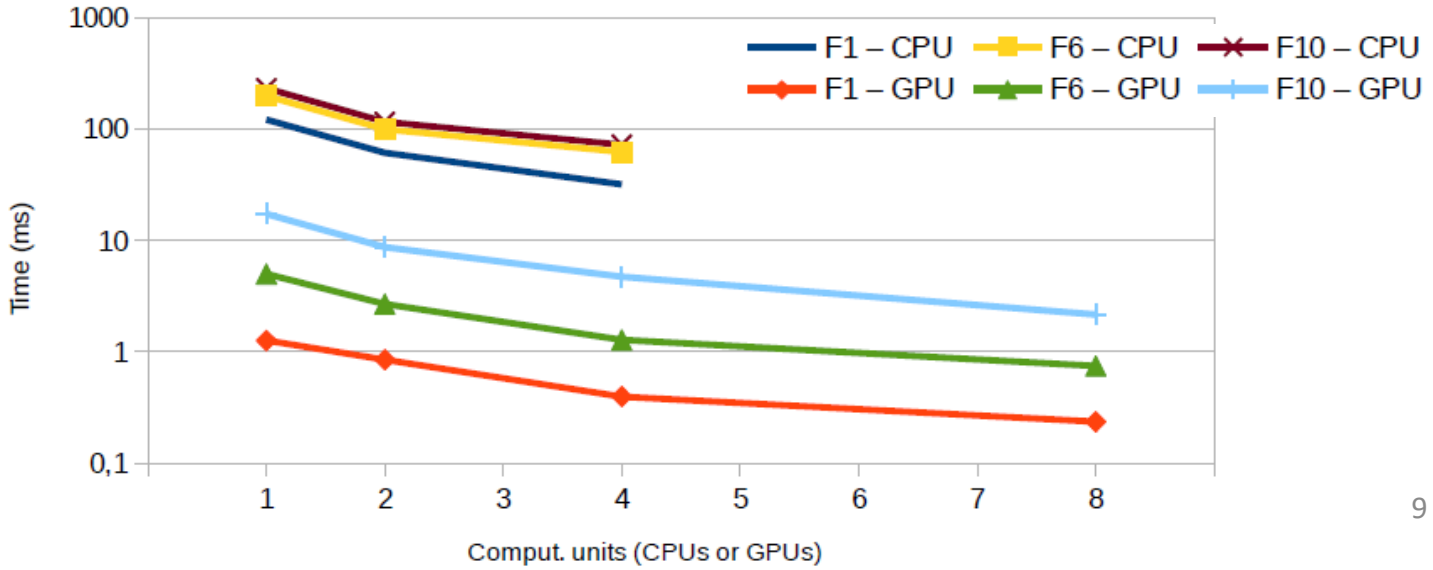
- Multiple-GPU parallelization of the fitness computation
  - Distribution of individuals
    - Simpler implementation, GPUs allocate a subset of individuals
    - Slow Genetic Operators due to GPU-to-GPU data transfers
  - Distribution of variables
    - Dimensions are mapped into multiple devices



- Partial fitness per GPU
- Minimum CPU-GPU sync
- Minimum CPU-GPU transfers
- No GPU-GPU sync / transfers
- Maximum throughput

- Experiment I: fitness function speedup
  - Evaluation time for F1, F6, F10 and 1 million dimensions

Evaluation	Function	1 CPU	2 CPUs	4 CPUs	1 GPU	2 GPUs	4 GPUs	8 GPUs
Time (ms)	F1	121.92	61.30	31.90	1.26	0.85	0.40	0.24
	F6	198.29	99.87	62.22	5.00	2.68	1.28	0.75
	F10	232.22	116.25	72.28	17.35	8.72	4.73	2.17
Speedup vs 1 CPU	F1	1	1.99	3.82	96.51	143.08	307.22	517.68
	F6	1	1.99	3.19	39.66	73.89	155.03	265.55
	F10	1	2.00	3.21	13.38	26.64	49.13	107.19



- Experiment II: fitness improvement (mean)

TABLE I
NGPUs-MA-SW-Chains vs its Predecessors ($D = 1,000$)

		F1	F2	F3	F4	F5
MA-SW-Chains		6.27e-13	1.24e+3	2.14e+1	4.96e+9	1.86e+6
	Std	5.34e-13	1.42e+2	4.52e-2	2.69e+9	3.66e+5
MA-SSW-Chains		8.18e-13	1.06e+3	2.03e+1	1.28e+10	1.34e+6
	Std	6.88e-13	1.22e+2	3.90e-2	3.79e+9	3.86e+5
NGPUs-MA-SW-Chains		**0**	**4.78e+0**	**2.00e+1**	**9.01e+6**	**1.05e+6**
	Std	**0**	**1.88e+0**	**0**	**7.10e+6**	**1.51e+5**
		F6	F7	F8	F9	F10
MA-SW-Chains		1.01e+6	3.69e+6	4.82e+13	5.38e+8	9.13e+7
	Std	1.38e+4	1.01e+6	9.92e+12	2.34e+8	8.18e+5
MA-SSW-Chains		1.05e+6	8.41e+7	1.44e+14	2.56e+8	9.35e+7
	Std	3.64e+3	2.68e+7	3.18e+13	1.45e+8	**2.38e+5**
NGPUs-MA-SW-Chains		**9.97e+5**	**2.27e+1**	**3.22e+11**	**1.19e+8**	**3.67e+6**
	Std	**1.44e+2**	**1.51e+1**	**3.37e+11**	**1.37e+7**	2.79e+6
		F11	F12	F13	F14	F15
MA-SW-Chains		9.24e+11	1.24e+3	1.89e+7	1.46e+8	5.17e+6
	Std	7.57e+9	**9.69e+1**	2.13e+6	1.75e+7	6.40e+5
MA-SSW-Chains		9.29e+11	1.34e+3	4.92e+9	3.78e+10	8.38e+6
	Std	9.54e+9	1.05e+2	1.63e+9	1.61e+10	1.52e+6
NGPUs-MA-SW-Chains		**1.67e+3**	**4.45e+2**	**1.66e+2**	**3.40e+3**	**5.39e+4**
	Std	**1.02e+3**	2.91e+2	**5.09e+1**	**2.62e+3**	**3.14e+3**

- Experiment III: extremely large number of dimensions

TABLE II
RESULTS IN 10^6 DIMENSIONS

	F1	F2	F3	F4	F5
NGPUs-MA-SW-Chains	**1.23e+13**	**1.85e+7**	**2.12e+1**	**8.91e+14**	**1.63e+10**
Std	9.02e+11	4.61e+4	**0**	**3.21e+13**	**7.08e+07**
NGPU Random Search	2.41e+14	5.97e+7	2.17e+1	1.32e+17	7.74e+10
Std	**3.92e+10**	**9.18e+3**	7.63e-5	9.94e+14	2.11e+8
	F6	F7	F8	F9	F10
NGPUs-MA-SW-Chains	1.08e+9	**2.76e+13**	**1.15e+19**	**1.43e+12**	9.80e+10
Std	3.77e+4	**2.48e+13**	**8.57e+17**	1.08e+11	1.49e+8
NGPU Random Search	**1.08e+9**	2.34e+21	8.10e+21	5.32e+12	**9.82e+10**
Std	**2.68e+4**	1.35e+20	2.98e+19	**1.83e+10**	**3.50e+6**
	F11	F12	F13	F14	F15
NGPUs-MA-SW-Chains	**9.72e+15**	**2.82e+13**	**1.26e+16**	**2.92e+16**	**3.50e+16**
Std	**1.47e+15**	2.21e+12	**3.87e+15**	**2.25e+16**	**2.04e+15**
NGPU Random Search	1.30e+23	1.99e+15	3.74e+22	2.91e+23	1.77e+19
Std	1.25e+22	**5.76e+11**	2.17e+21	1.84e+22	5.07e+15

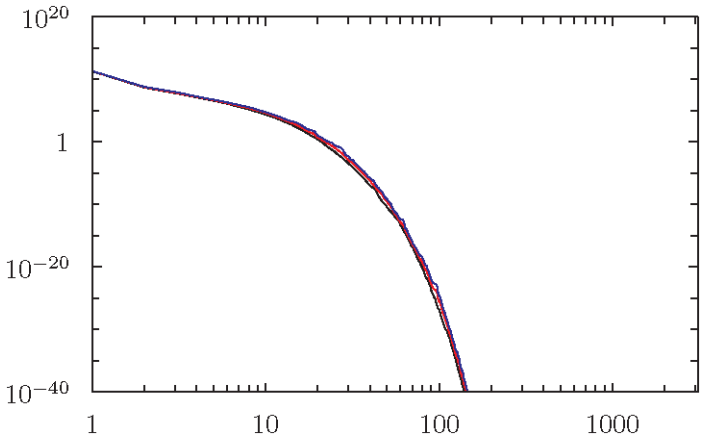
- Experiment III: extremely large number of dimensions

TABLE IV
RESULTS IN 10^8 DIMENSIONS

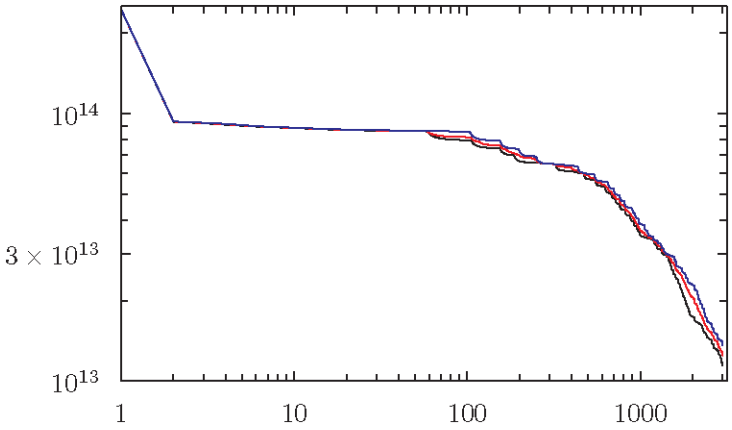
	F1	F2	F3	F4	F5
NGPUs-MA-SW-Chains	**8.42e+15**	**1.95e+9**	2.17e+1	**2.44e+18**	**2.55e+12**
Std	8.65e+14	7.59e+6	8.36e-3	1.73e+17	1.21e+10
NGPU Random Search	2.43e+16	6.00e+9	**2.17e+1**	1.44e+19	8.09e+12
Std	**2.64e+11**	**4.38e+4**	**0**	**8.24e+15**	**2.91e+9**
	F6	F7	F8	F9	F10
NGPUs-MA-SW-Chains	1.08e+11	**1.81e+19**	**1.95e+23**	**2.07e+14**	**9.74e+12**
Std	3.42e+6	**4.31e+21**	2.15e+22	1.52e+13	1.22e+10
NGPU Random Search	**1.08e+11**	1.39e+24	8.86e+23	5.59e+14	9.82e+12
Std	**3.00e+5**	2.51e+22	**5.08e+20**	**7.48e+10**	**5.46e+7**
	F11	F12	F13	F14	F15
NGPUs-MA-SW-Chains	**7.46e+21**	**4.59e+16**	**2.39e+21**	**9.67e+22**	**2.68e+19**
Std	**1.96e+21**	3.29e+15	**1.33e+21**	**6.16e+22**	1.80e+18
NGPU Random Search	1.70e+26	2.00e+17	1.32e+25	1.56e+26	7.65e+19
Std	2.43e+24	**4.90e+11**	2.13e+23	1.53e+24	**5.56e+16**

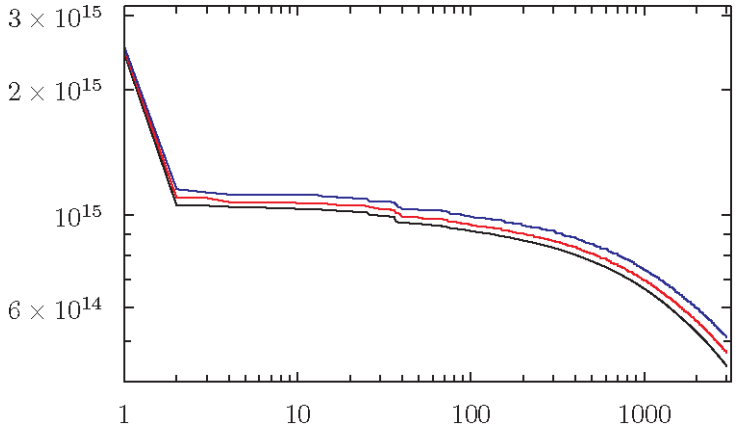
- Experiment IV: convergence F1 function

1,000 dimensions

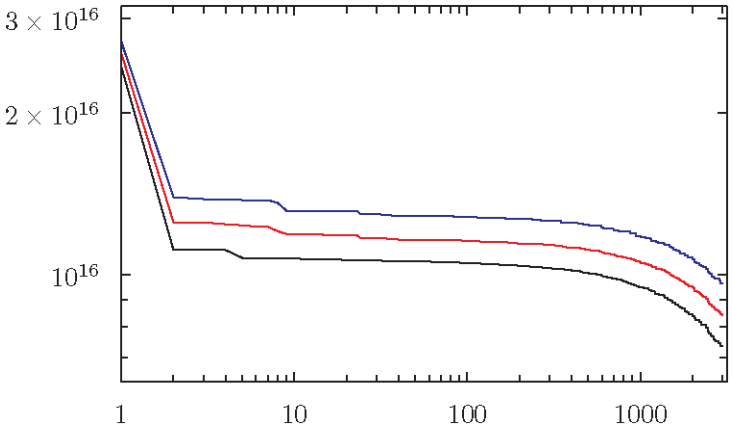


1 million dimensions





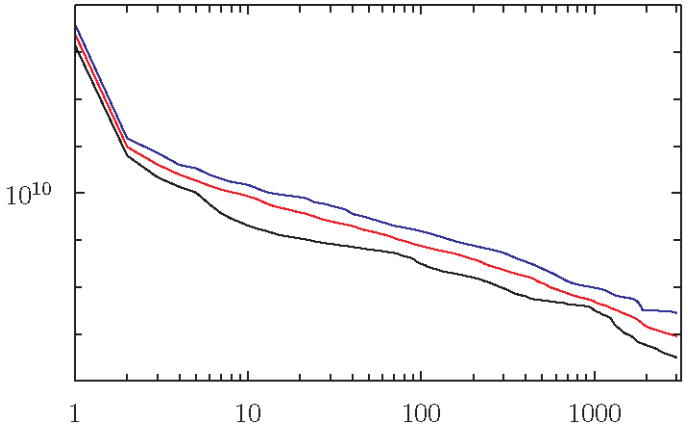
10 million dimensions



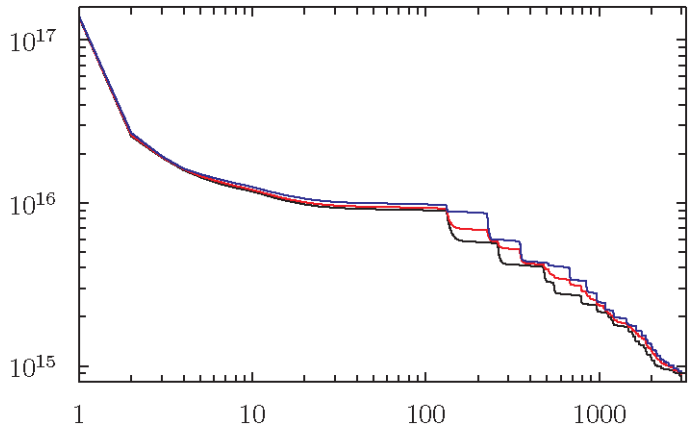
100 million dimensions

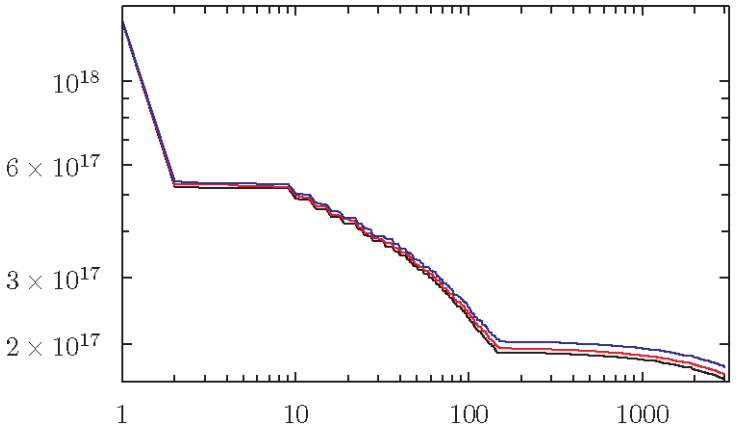
- Experiment IV: convergence F4 function

1,000 dimensions

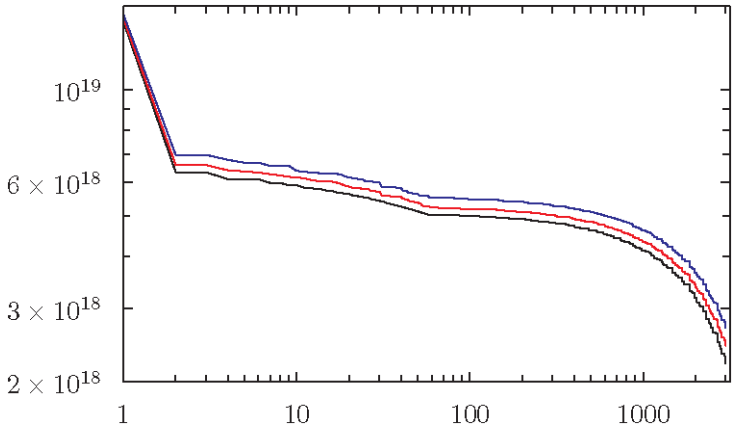


1 million dimensions





10 million dimensions



100 million dimensions

- Conclusions and future work
  - Multiple GPUs speed up fitness computation and evolution on extremely high-dimensional functions with components
  - Distribution of the dimensions provides an efficient and scalable approach for large-scale global optimization using multiple GPUs
  - GPUs may be employed to generate multiple solutions per iteration, leading to better results (not equal number of FEs)
  - 1 billion-scale dimensionality challenge
    - NVIDIA DGX-1 system, 8 GPUs and 128 GB memory
  - Scaling IEEE CEC 2015 Competition winner:
    - MOS (Multiple Offspring Sampling)
  - Need for real-world applications / functions!!
  - Hadoop / Spark implementation?

IEEE World Congress on Computational Intelligence 2016

100 Million Dimensions Large-Scale Global Optimization Using Distributed GPU Computing

Alberto Cano and Carlos García-Martínez

Alberto Cano
Assistant Professor
Department of Computer Science
Virginia Commonwealth University, USA
acano@vcu.edu

IEEE WCCI 2016
Vancouver Canada