Improving the Energy Efficiency of Evolutionary Multi-Objective Algorithms

J.J. Moreno[1], G. Ortega[1], E. Filatovas[2], J.A. Martínez[1] and E.M. Garzón[1]

December 15, 2016

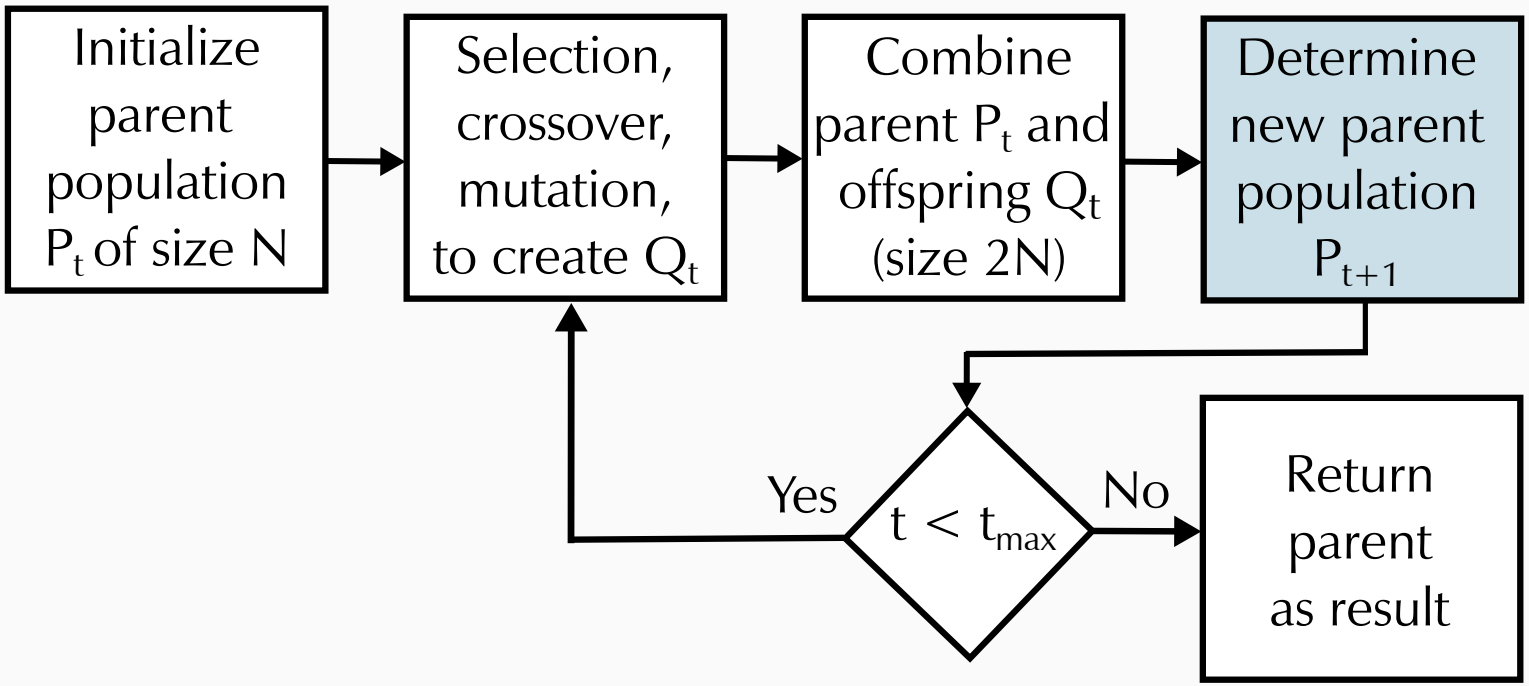
[1] Informatics Department, University of Almería, Almería, Spain
[2] Institute of Mathematics and Informatics, Vilnius University, Vilnius, Lithuania

Outline

1. NSGA-II: Non-Dominated Sorting Genetic Algorithm II.
2. GPU implementation of the NDS procedure of NSGA-II.
3. Evaluation tools and methods.
4. Results, conclusions and future work.

NSGA-II

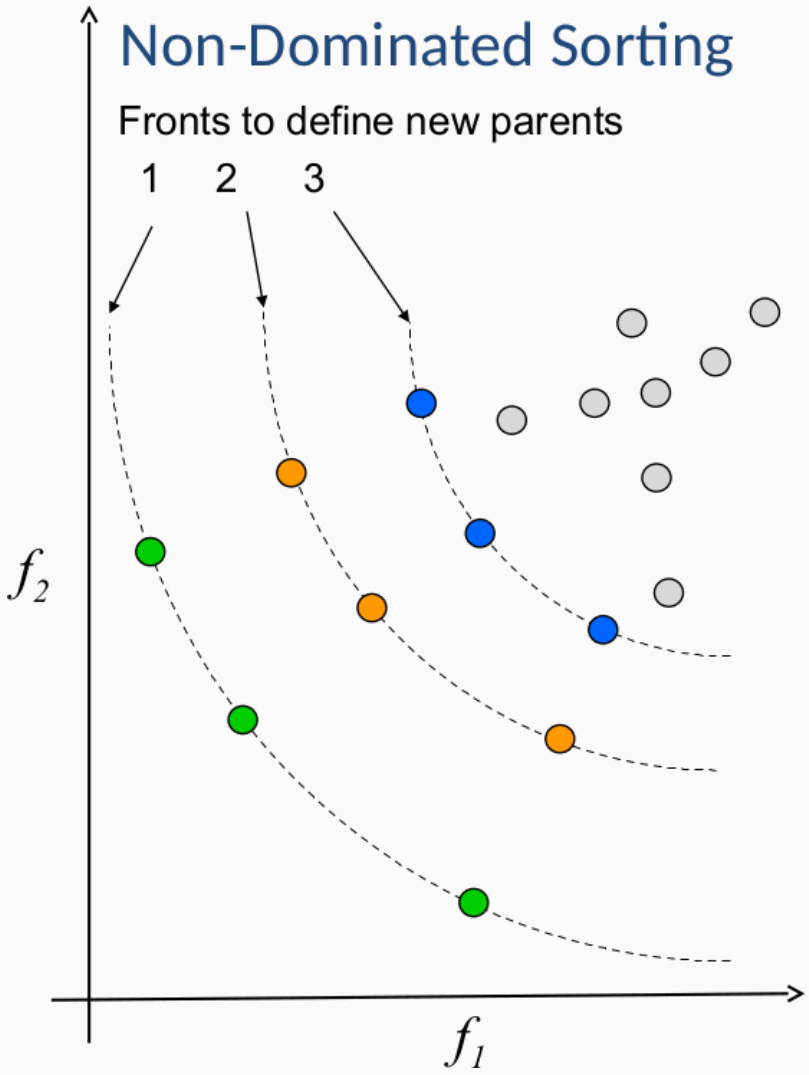
NSGA-II flowchart



P_{t+1} = Non-dominated sorting + crowding distance sorting

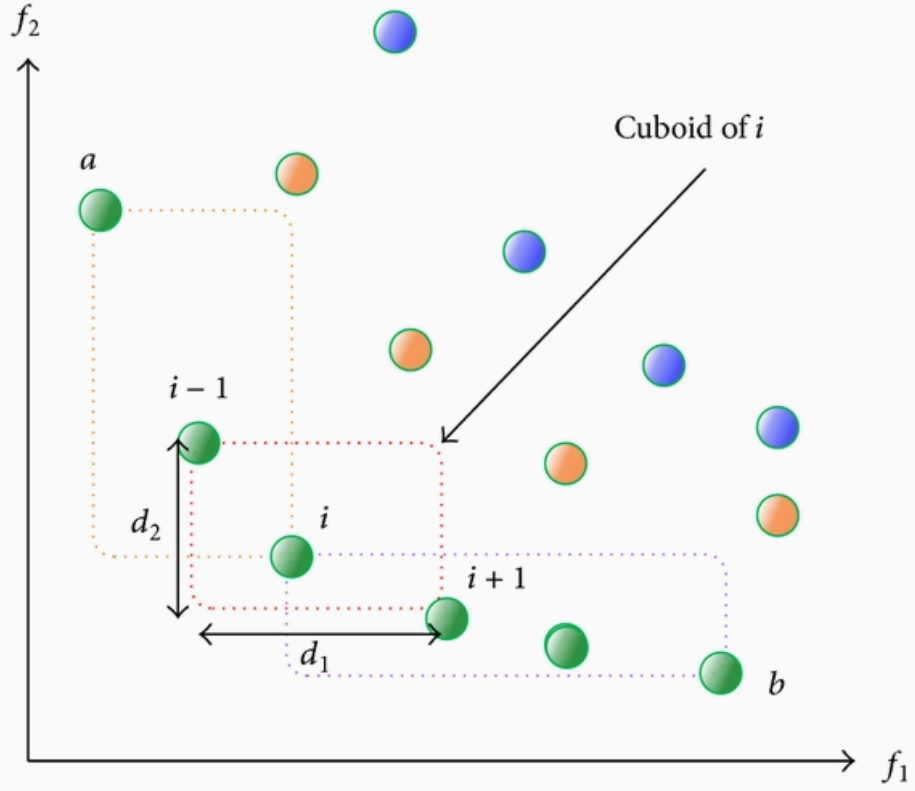
Non-dominated sorting

- An individual is non-dominated when there is no other individual better than it for all objective functions.
- The set of non-dominated individuals is called a front.

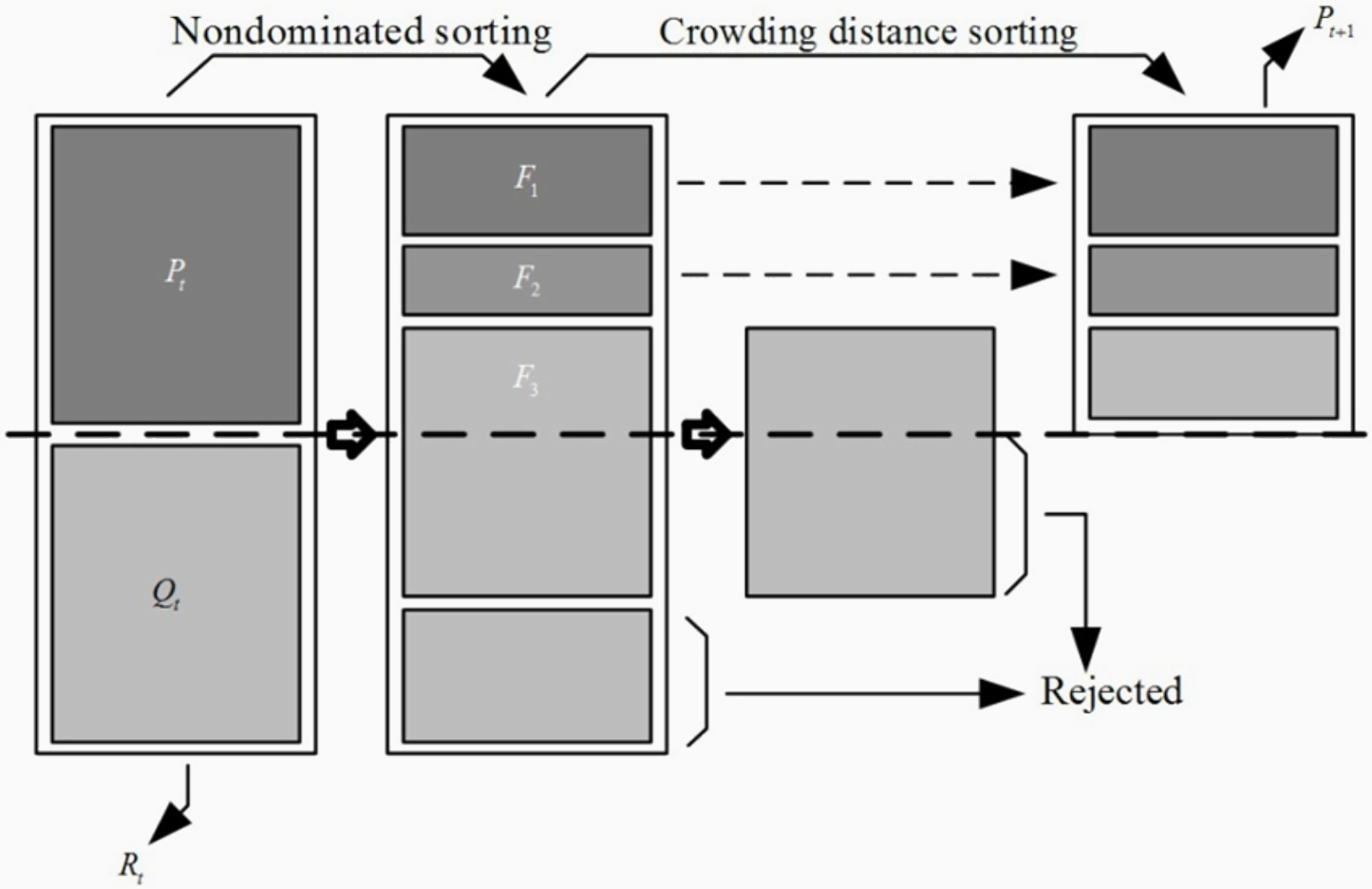


Crowding distance sorting

- Used to ensure diversity in the population.
- Measures how close each individual is to its front neighbors.

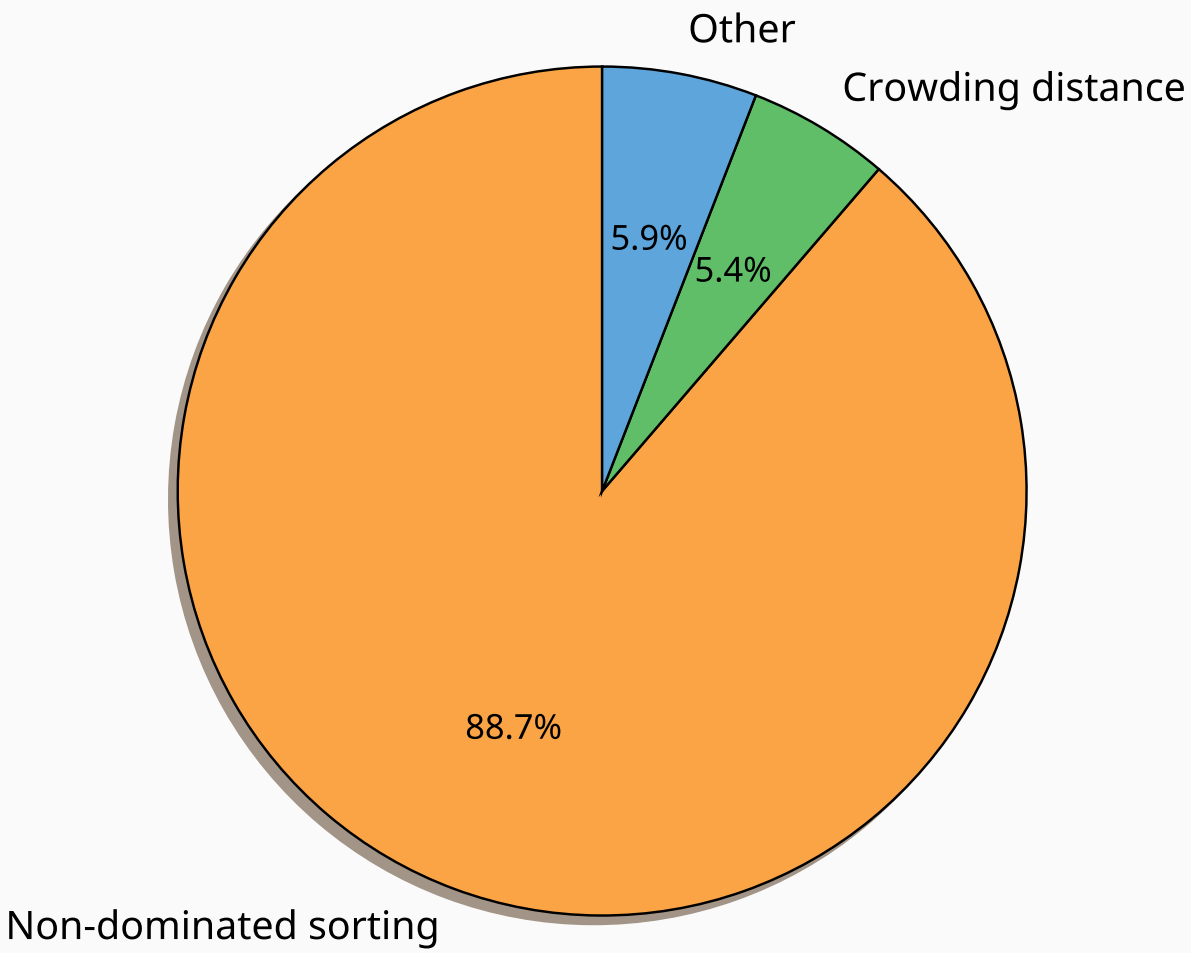


Selection of the best individuals



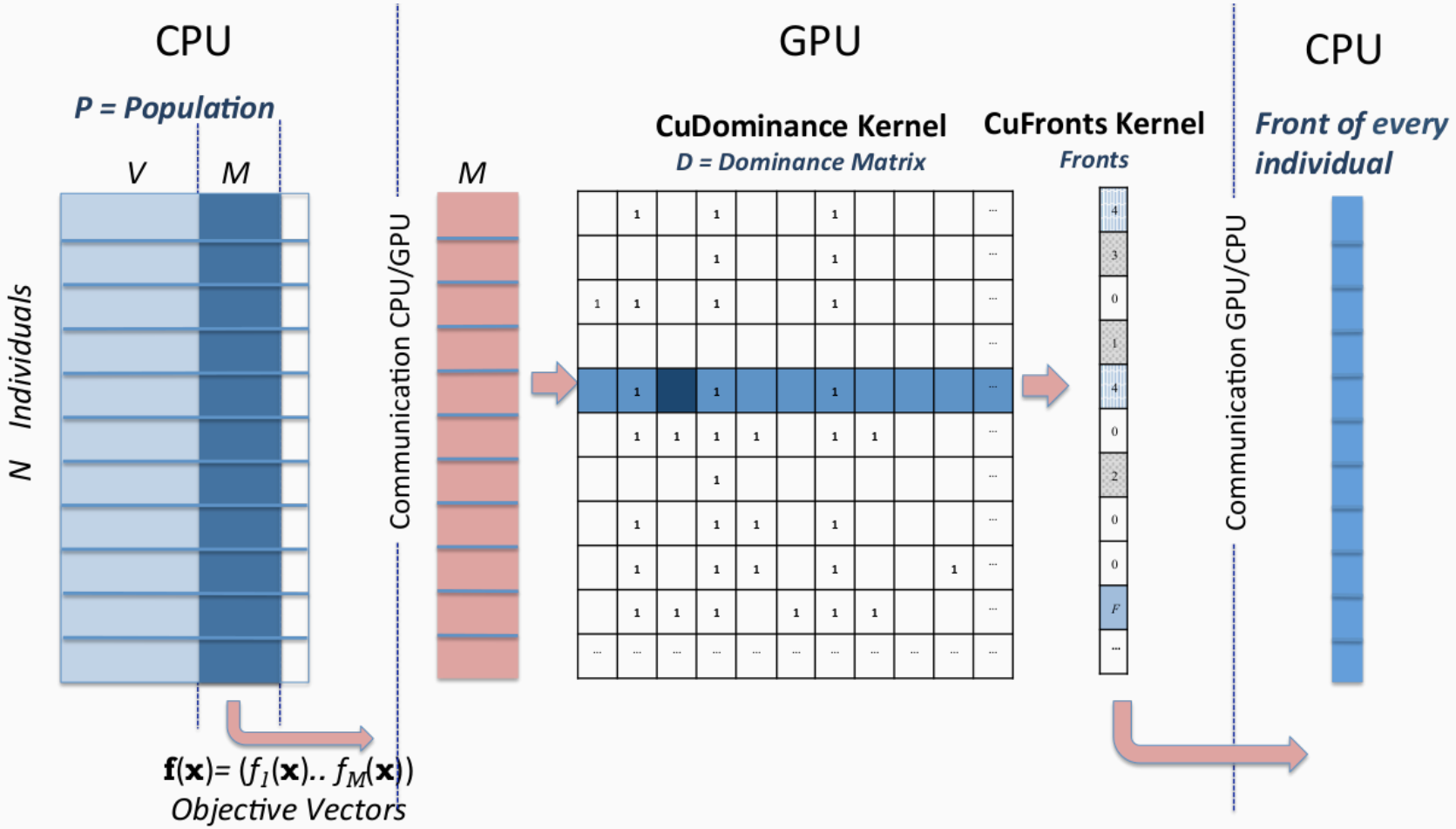
We return the best *N* individuals from a population of size 2*N*

NSGA-II runtime analysis

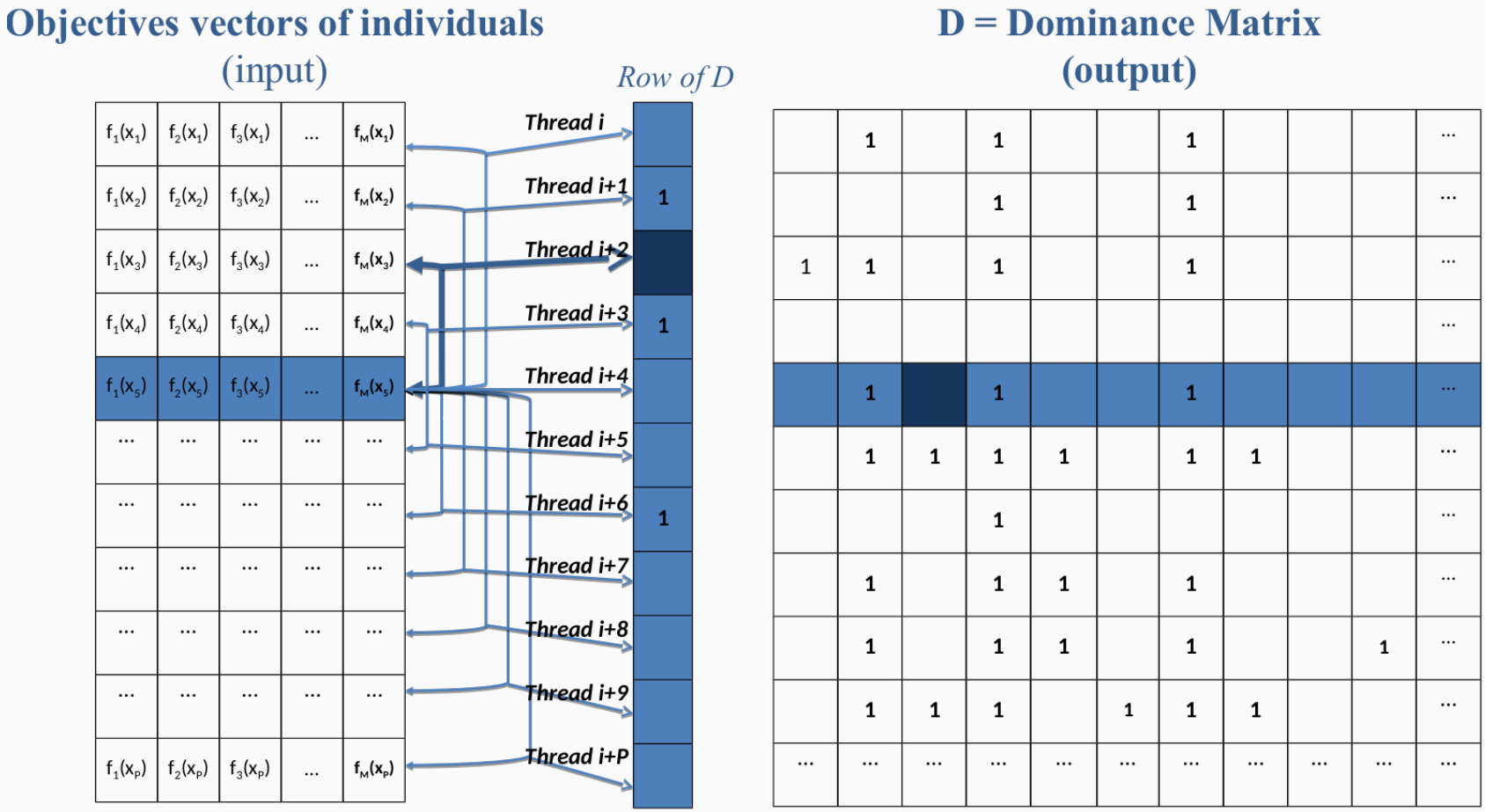


Efficient NDS on GPU

Outline

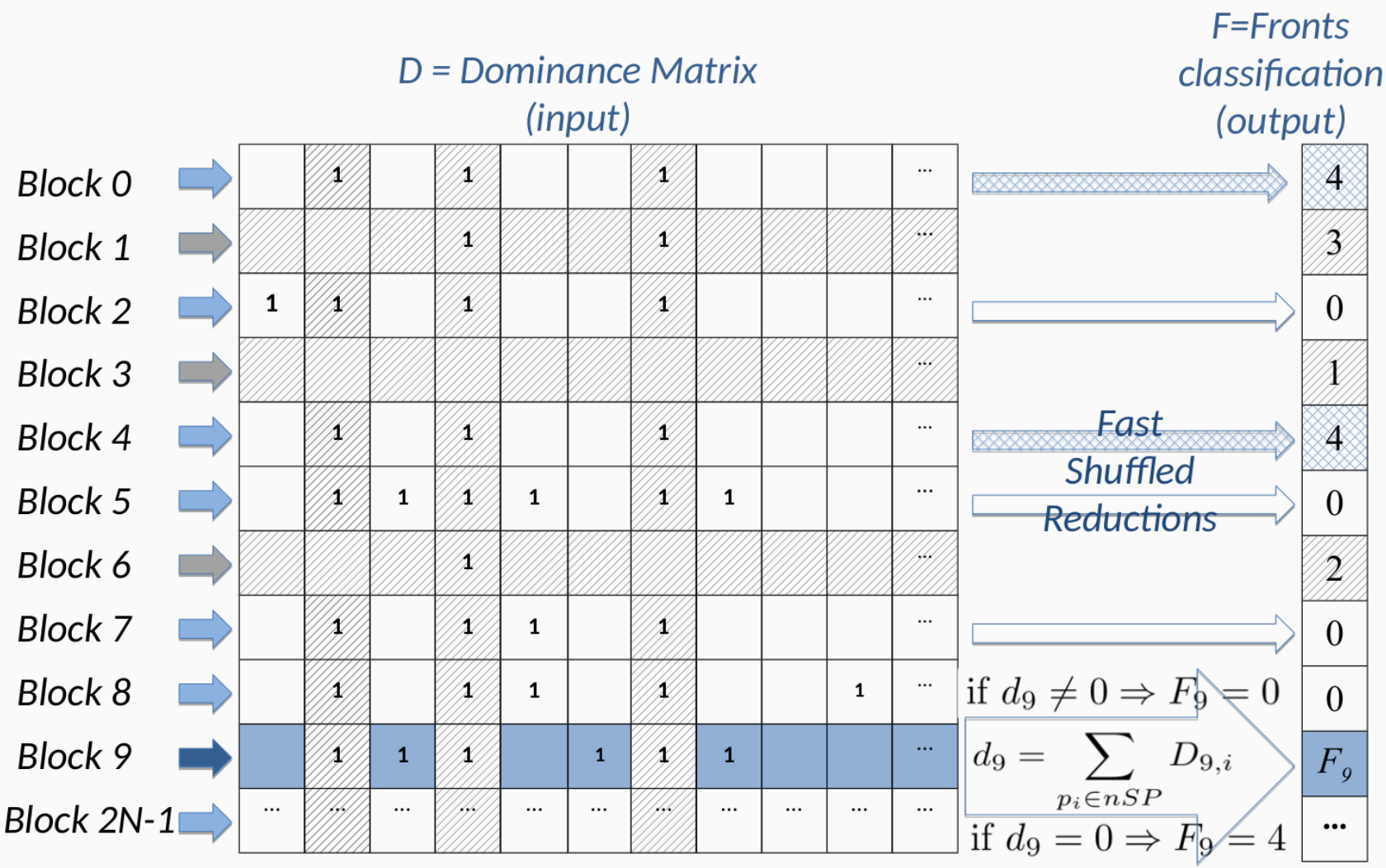

CPU
P = Population
V
M
N Individuals
f(x)= (f1(x).. fM(x))
Objective Vectors
Communication CPU/GPU
M
GPU
CuDominance Kernel
D = Dominance Matrix
CuFronts Kernel
Fronts
CPU
Front of every individual
Communication GPU/CPU

CuDominance kernel



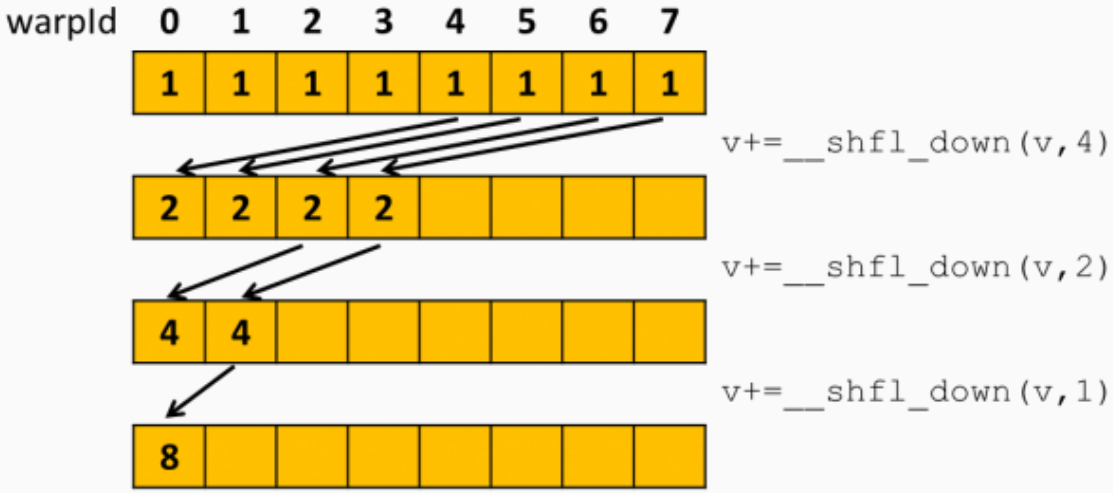
$D_{ij} = 1 \rightarrow P_j$ dominates P_i

CuFronts kernel



Warp Shuffle instructions

- Introduced in the Kepler microarchitecture.
- Enables a thread to directly read a register from another thread in the same warp.
- Four intrinsics: `__shfl()`, `__shfl_down()`, `__shfl_up()`, `__shfl_xor()`.



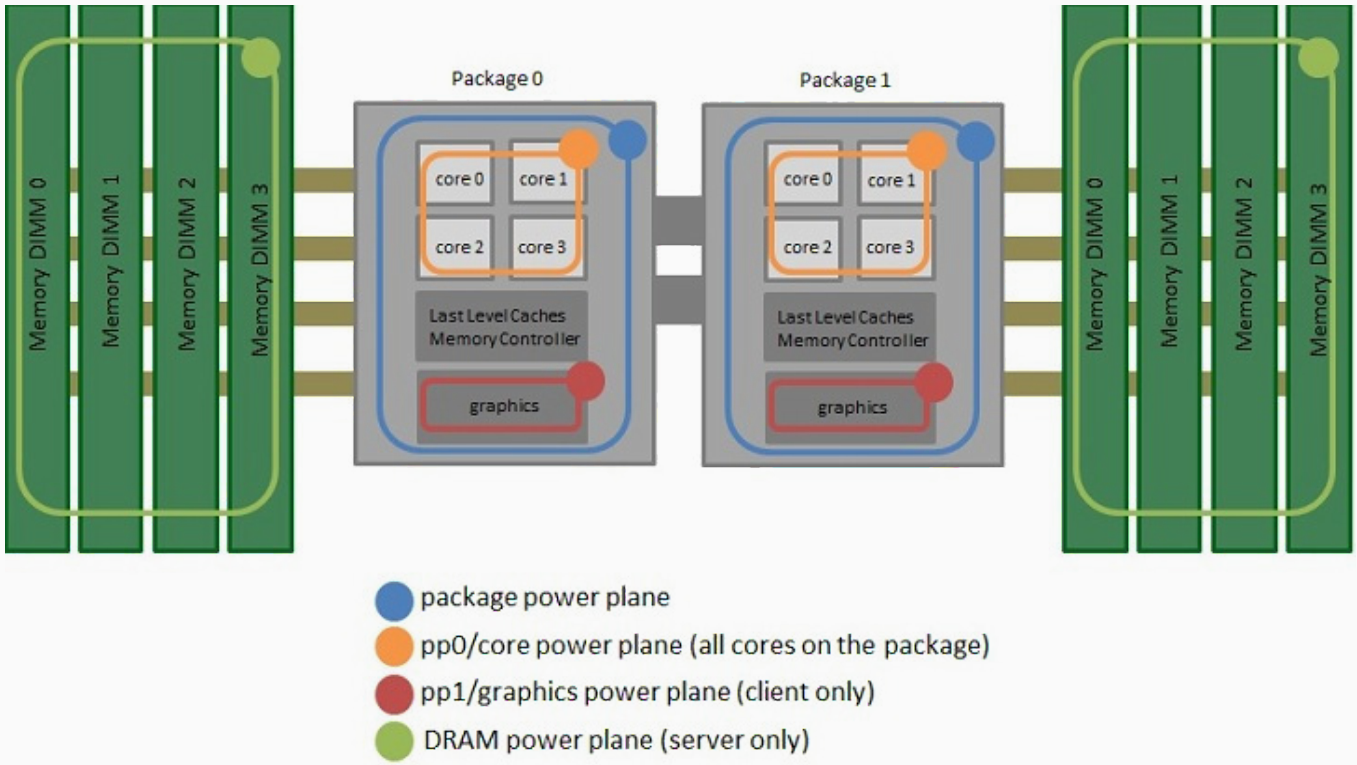
Evaluation methods and tools

Test platform

- **CPU:** 2 x Intel Xeon CPU E5-2620 v3 @ 2.40 GHz.
- **GPU:** 2 x NVIDIA Tesla K80.
- **RAM:** 64 GiB DDR3 @ 2133 MHz.
- **OS:** Ubuntu GNU/Linux 16.04.1 LTS.
- **SDK:** CUDA 8.0.

Energy measurement on Intel processors

- Using the Intel Running Average Power Limit interface.
- Available since the Sandy Bridge microarchitecture.
- Provides energy consumed in Joules.



Energy measurement on NVIDIA GPUs

- Using the NVIDIA Management Library.
- Available since the Kepler microarchitecture for the Tesla and Quadro GPU families.
- Provides instant power in Watts with an error of $\pm$ 5 W.

DTLZ test problems

- Designed for evaluating multi-objective algorithms.
- Allow an arbitrary number of objective functions.

DTLZ2

$$
\begin{aligned}
&\min f_1(\mathbf{x}) = (1 + g(x)) \prod_{i=1}^{M-1} cos(\frac{x_i \pi}{2}) \\
&\min f_2(\mathbf{x}) = (1 + g(x)) sin(\frac{x_{M-1} \pi}{2}) \prod_{i=1}^{M-2} cos(\frac{x_i \pi}{2}) \\
&\dots \\
&\min f_l(\mathbf{x}) = (1 + g(x)) sin(\frac{x_{M-l+1} \pi}{2}) \prod_{i=1}^{M-l} cos(\frac{x_i \pi}{2}) \\
&\dots \\
&\min f_M(\mathbf{x}) = (1 + g(x)) sin(\frac{x_1 \pi}{2}) \\
&g(\mathbf{x}) = \sum_{i=M}^{n} (x_i - 0.5)^2, x_i \in [0, 1]
\end{aligned}
\tag{1}
$$

DTLZ7

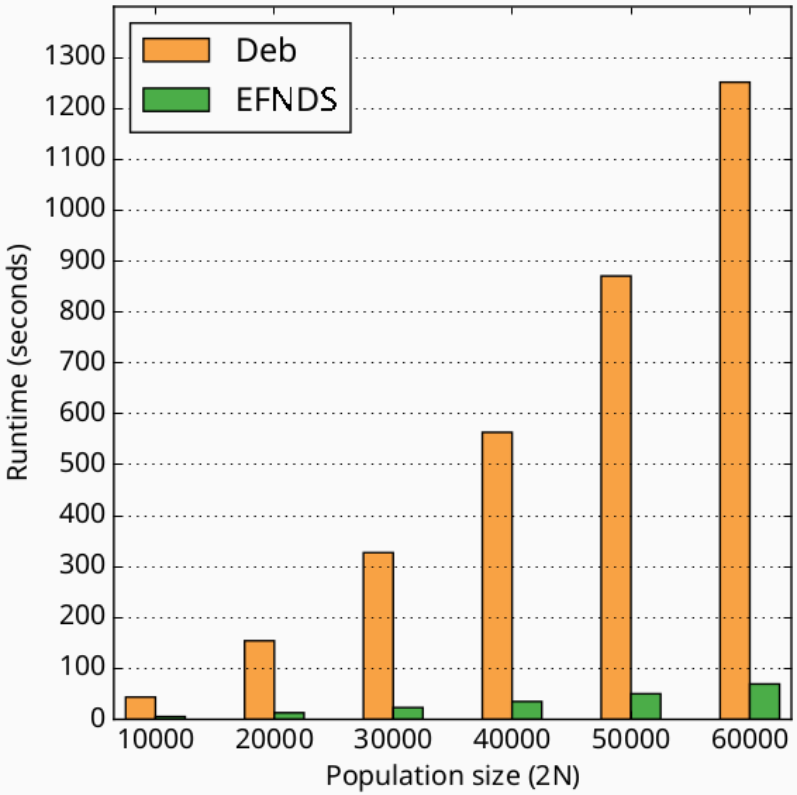
$$
\begin{aligned}
&\min \quad f_1(\mathbf{x}) = x_1 \\
&\min \quad f_2(\mathbf{x}) = x_2 \\
&\dots \\
&\min \quad f_{M-1}(\mathbf{x}) = x_{M-1} \\
&\min \quad f_M(\mathbf{x}) = (1 + g(x_M)) h(f_1, f_2, \dots, f_{M-1}, g) \\
&\text{where} \quad g(x_M) = 1 + \frac{9}{|x_M|} \sum_{x_i \in x_M} x_i \\
&\qquad h(f_1, f_2, \dots, f_{M-1}, g) = \\
&\qquad M - \sum_{i=1}^{M-1} \left[\frac{f_1}{1+g} (1 + \sin(3\pi f_i)) \right] \\
&\text{subject to} \quad 0 \le x_1 \le 1, \quad \text{for } i = 1, 2, \dots, n
\end{aligned}
\tag{2}
$$

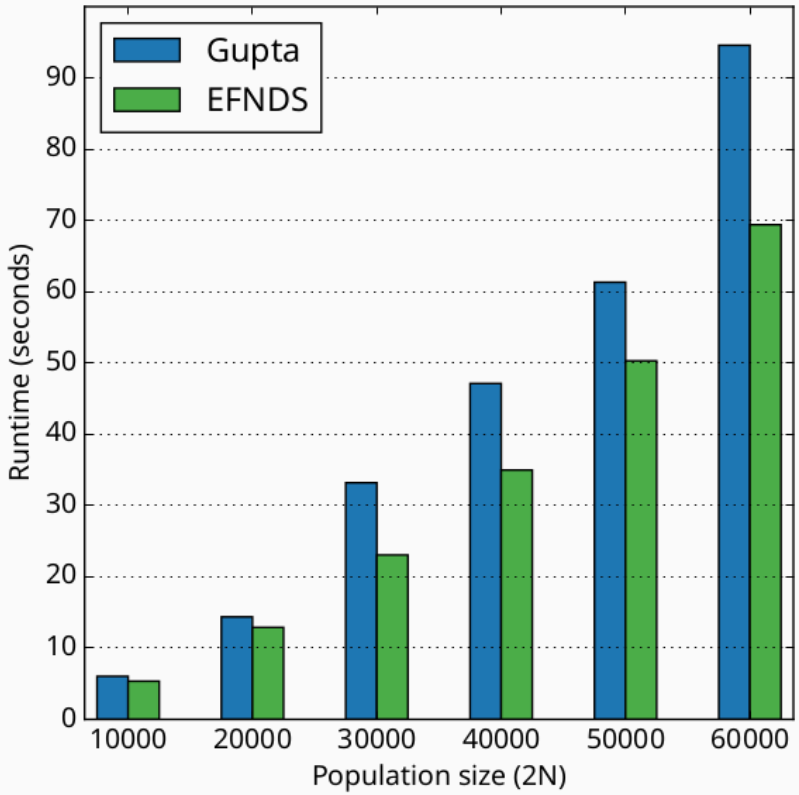
Results, conclusions and future work

Run configurations

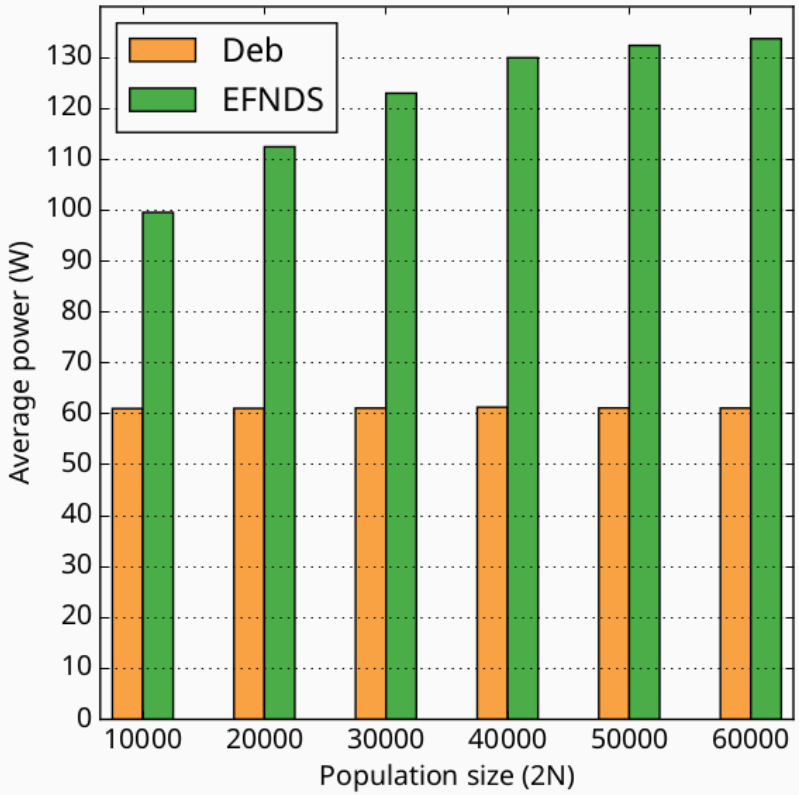
- Three implementations:
  1. Sequential implementation by Kalyanmoy Deb.
  2. GPU implementation by Samarth Gupta.
  3. Our GPU implementation.
- Two test problems: DTLZ2 and DTLZ7.
- 5, 10 and 15 objective functions.
- 50 generations.
- Varying population sizes.

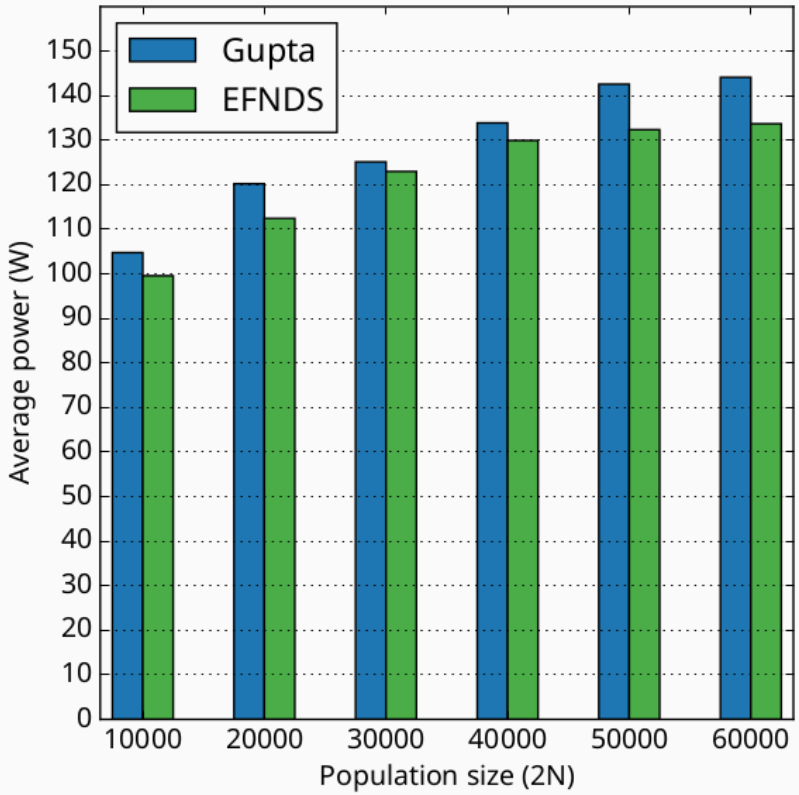
Runtime of DTLZ7 with 5 objectives



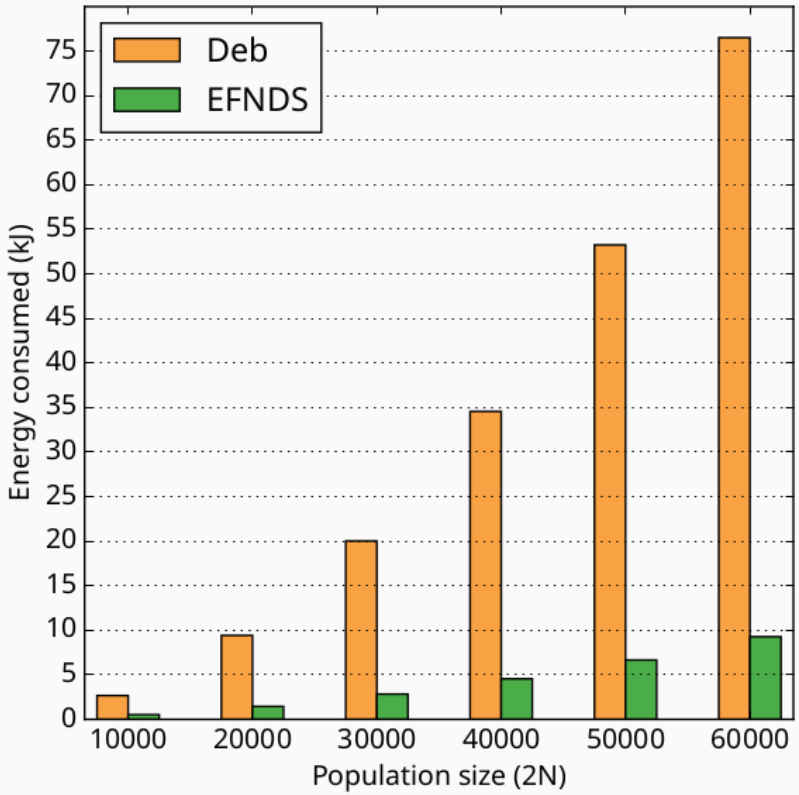


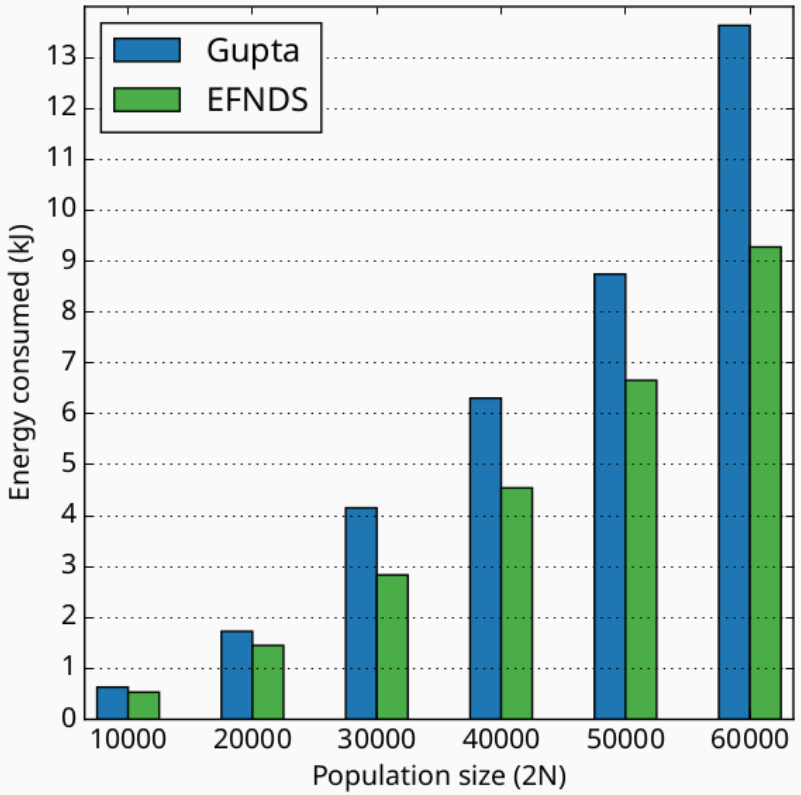
Average power of DTLZ7 with 5 objectives





Energy consumed by DTLZ7 with 5 objectives





Conclusions

- We have proposed a GPU implementation of the Non-Dominated Sorting procedure of the NSGA-II algorithm.
- Our evaluation has shown that this implementation is faster and more energy efficient than the other proposals for this platform and these test problems.

Future work

- Multi-core and Multi-GPU implementation.
- Evaluate our proposal in other platforms.
- Use real multi-objective problems.

Thanks!