Asynchronous Parallel Stochastic Global Optimization using Radial Basis Functions

David Eriksson

Center for Applied Mathematics
Cornell University

dme65@cornell.edu

October 24, 2017

Joint work with David Bindel and Christine Shoemaker

Global optimization problem (GOP)

$$\text{Find } x^* \in \Omega \text{ such that } f(x^*) \leq f(x),\ \forall x \in \Omega$$

- $f : \Omega \to \mathbb{R}$ continuous, computationally expensive, and black-box
- $\Omega \subset \mathbb{R}^d$ is a hypercube
- Evaluating the model may take several hours or days
- Common examples are PDE models describing physical processes

Difficulty with popular approaches for global optimization

- (Multi-start) Gradient based optimizers:
  - **Examples:** Gradient descent, quasi-Newton methods
  - **Problem:** Hard to obtain (accurate) derivatives, multi-modality
  - Tricky to choose step size for finite differences
  - Finite differences are expensive in higher dimensions
- (Multi-start) Derivative-free methods:
  - **Examples:** Nelder-Mead, pattern search
  - **Problem:** Slow convergence, multi-modality, ignores smoothness
- Heuristic methods:
  - **Examples:** Genetic algorithm, simulated annealing
  - **Problem:** Require a large number of evaluations

Surrogate optimization

- Use a surrogate $\hat{f}$ (- - -) to approximate f (——)
- The surrogate enables cheap function value predictions
- **Main idea:** Solve auxiliary problem, evaluate, fit surrogate, repeat

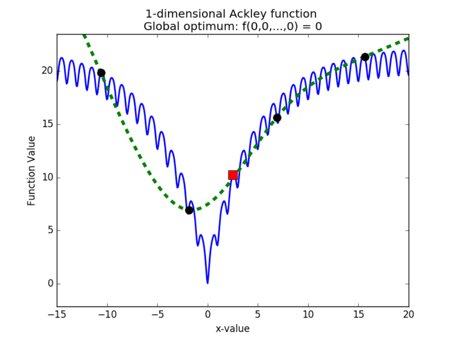
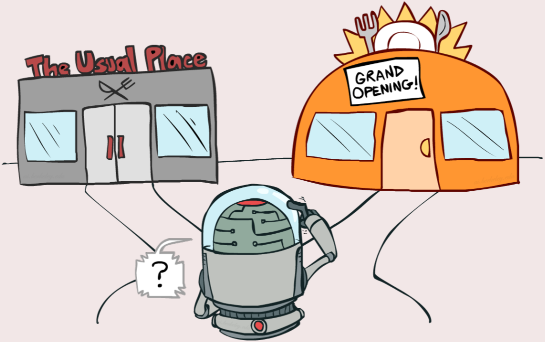


Figure: (●) Evaluated points, (■) next evaluation.

Exploration vs exploitation

A successful method needs to balance exploration and exploitation

- **Exploration:** Evaluate in unexplored regions
- **Exploitation:** Improve good solutions



Radial basis function interpolation

$$s_{f,X}(x) = \sum_{j=1}^{n} \lambda_j \varphi(\|x - x_j\|) + p(x)$$

- $p(x) = \sum_{j=1}^{m} c_j \pi_j(x)$ a polynomial of degree k
- Interpolation constraints:

$$s(x_j) = f(x_j), \qquad j = 1, \ldots, n$$

- Discrete orthogonality:

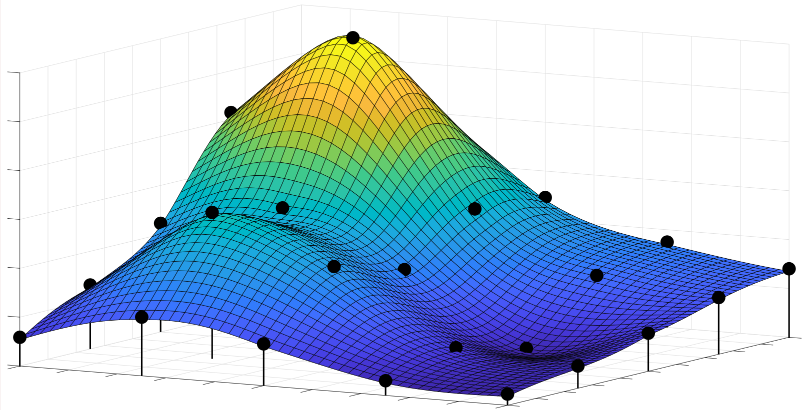
$$\sum_{j=1}^{n} \lambda_j q(x_j) = 0, \qquad \forall \text{poly } q \text{ of deg} \leq k$$

- Need to solve

$$\begin{bmatrix} \Phi & P \\ P^T & 0 \end{bmatrix} \begin{bmatrix} \lambda \\ c \end{bmatrix} = \begin{bmatrix} f_X \\ 0 \end{bmatrix}$$

where $\Phi_{ij} = \varphi(\|x_i - x_j\|)$, $P_{ij} = \pi_j(x_i)$

Radial basis function interpolation



Stochastic Radial Basis Function (SRBF) method

- Uses a radial basis function to approximate objective
- Generate a set of candidate points Λ
- Each candidate point is a $\mathcal{N}(0, \sigma^2)$ perturbation of best solution
- Sampling radius σ is adjusted based on progress
- Auxiliary problem:

$$\min_{x \in \Lambda} \left[\lambda \frac{s(x) - \min_{y \in \Lambda} s(y)}{\max_{y \in \Lambda} s(y) - \min_{y \in \Lambda} s(y)} + (1 - \lambda) \left(\frac{\max_{y \in \Lambda} d_X(y) - d_X(x)}{\max_{y \in \Lambda} d_X(y) - \min_{y \in \Lambda} d_X(y)} \right) \right]$$

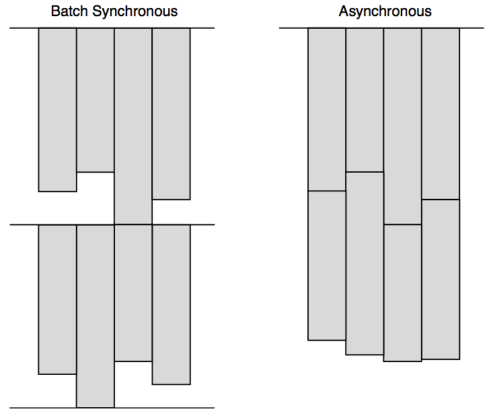
where $d_X(y) = \min_{x \in X} \|x - y\|$, $\lambda \in [0, 1]$.

- $(\lambda = 0)$ favors large minimum distance to evaluated points
- $(\lambda = 1)$ favors small function value prediction

Parallelism

Running function evaluations in parallel:

1. Batch synchronous parallel
2. Asynchronous parallel



Parallelism

- Synchronous parallel assumes:
  1. Computational resources are homogeneous
  2. Evaluation time independent of input
- Examples of heterogeneous resources:
  - Mixture of CPU/GPU
  - Clouds (e.g., "stragglers" in MapReduce)
- Examples of input dependent evaluation time:
  - Adaptive meshes
  - Iterative solver (Krylov, bisection, etc.)
  - Early termination

POAP and pySOT

POAP (Plumbing for Optimization with Asynchronous Parallelism)

- Available at: `https://github.com/dbindel/POAP`
- Framework for building asynchronous optimization strategies

pySOT (Python Surrogate Optimization Toolbox)

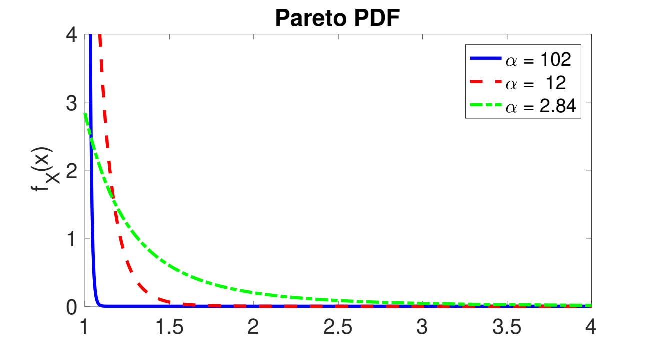
- Available at: `https://github.com/dme65/pySOT`
- Surrogate optimization strategies implemented in POAP
- A great test-suite for doing head-to-head comparisons
- Has been cited in work on:
  - Groundwater flow calibration for the Umatilla Chemical Depot
  - Calibration of a geothermal reservoir model
  - Hyper-parameter optimization of deep neural networks

Questions to answer

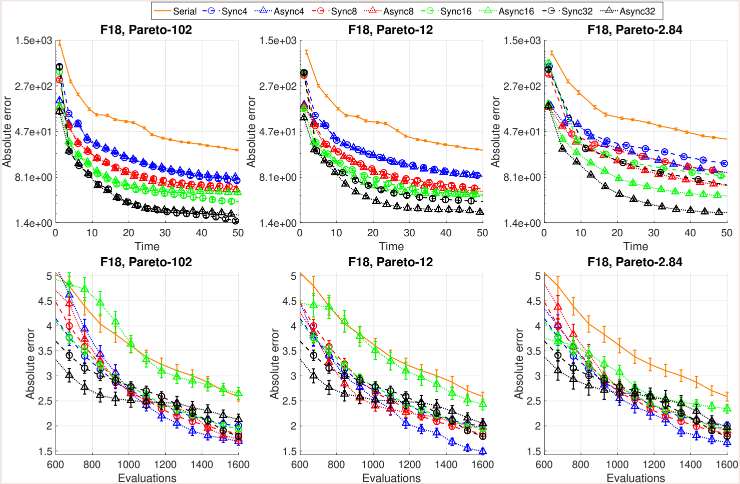
1. How do we choose between asynchrony and synchrony?
2. What is the tradeoff between information and idle time?
3. What is the effect of parallelism?

Experimental setup for test problems

- Use SRBF with 1, 4, 8, 16, and 32 workers
- 10-dimensional F15-F24 from the BBOB test suite
- Draw eval time from Pareto distribution: $f_X(x) = \frac{\alpha}{x^{1+\alpha}} \mathbf{1}_{[1,\infty)}(x)$
- Vary $\alpha \in \{102, 12, 2.84\}$ to achieve different tail behaviors
- Corresponds to standard deviations 0.01, 0.1, and 1

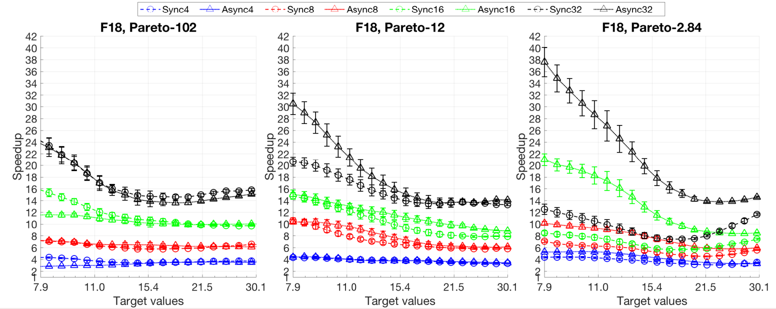


Progress comparison for F18



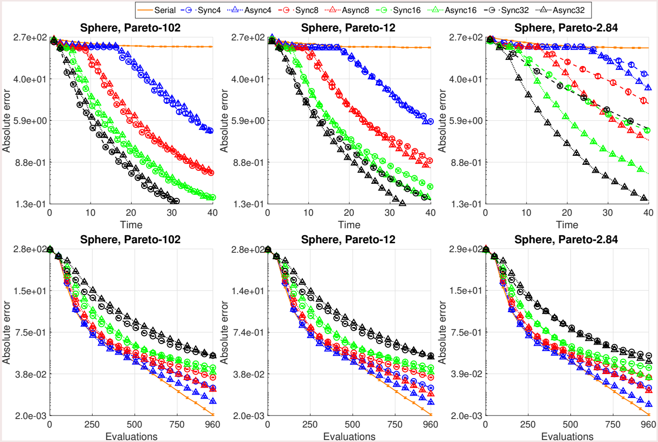
Relative speedup for F18

- Relative speedup: $\tilde{S}(p) = \dfrac{\text{Execution time for serial algorithm}}{\text{Execution time for parallel algorithm with } p \text{ processors}}$
- Computed over intersection of ranges from all runs



Progress comparison for unimodal function

- Consider the sphere function: $f(x) = \sum_{j=1}^{30} x_j^2$



Answers to questions

1. How do we choose between asynchrony and synchrony?
    - Asynchrony is the best choice on multimodal problems
    - Best on all problems in large variance case
    - In small variance case asynchrony better vs time on
        - 7/10 problems with 4 processors
        - 6/10 problems with 8 processors
        - 5/10 problems with 16 processors
        - 5/10 problems with 32 processors
2. What is the tradeoff between information and idle time?
    - Idle time more important than information for multimodal problems
    - Serial not necessarily best vs #evals in multimodal case
    - Serial best vs #evals for unimodal problems
3. What is the effect of parallelism?
    - Helps with exploration
    - Improves results vs time

Thank you!