Single–Run Simulation Optimization through Time Dilation and Optimal Computing Budget Allocation

Giulia Pedrielli(*),
Yinchao Zhu, and Loo Hay Lee(**)

(*) Centre for Maritime Studies, National University of Singapore,
cmsgp@nus.edu.sg ,
(**) Department of Industrial & Systems Engineering, National University of Singapore

June 3rd, 2015

Outline

Problem Statement: Simulation–Optimization of DESs

"[...] optimization is the most significant new simulation technology [...]." (Law and Mc Comas, 2002)

$$
\begin{aligned}
P_o \quad & \text{minimize} \quad f_o(\boldsymbol{x}) = \mathbb{E}\{y_o(\boldsymbol{x}, \omega)\} = 0 \\
& \text{subject to} \\
& \qquad f_j(\boldsymbol{x}) = \mathbb{E}\{y_j(\boldsymbol{x}, \omega)\} = 0, \quad j = 1, \ldots, p < d \\
& \qquad f_j(\boldsymbol{x}) = \mathbb{E}\{y_j(\boldsymbol{x}, \omega)\} \geq 0, \quad j = p+1, \ldots, p+q \\
& \qquad \boldsymbol{x} \in \mathbb{X}^d
\end{aligned}
$$

Sequential Stochastic Selection

Objective: Given $\mathbb{X}^d$ is a **set of candidate solutions**, identify the best system using an iterative procedure constrained by a *finite number of function evaluations* T.

S^3 Method for DES

1. Single Run (largely developed in stochastic search) → use the event frequency to reduce the variance where useful. This has led to *TIME DILATION* (Hyden et al. 2000);
2. Multiple Replicates (typical in R&S approach) → use replications allocation to reduce variance where useful. This has led to *OPTIMAL COMPUTING BUDGET ALLOCATION* (Chen et al. 2000).

S^3 Method for DES

1. Single Run (largely developed in stochastic search) → use the event frequency to reduce the variance where useful. This has led to *TIME DILATION* (Hyden et al. 2000);
2. Multiple Replicates (typical in R&S approach) → use replications allocation to reduce variance where useful. This has led to *OPTIMAL COMPUTING BUDGET ALLOCATION* (Chen et al. 2000).

Objective

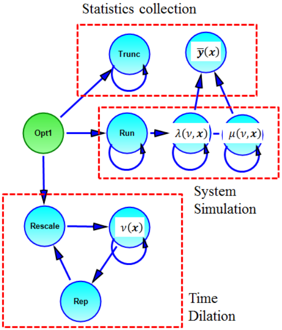
Propose an *integrated* solution for **DES** simulation & Optimization within the class of single run algorithms **merging OCBA and TIME DILATION**.

Time Dilation: the *Work model*

Basic Idea: competing systems in an optimization environment can be simulated simultaneously and get their time scales dilated based on their performance.

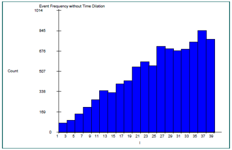
Time Dilation: the *Work model*

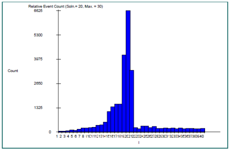
Basic Idea: competing systems in an optimization environment can be simulated simultaneously and get their time scales dilated based on their performance.



At predefined time points we update each system time scale:

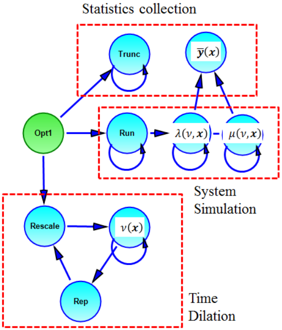
$$1/\lambda \leftarrow \nu/\lambda',\ 1/\mu \leftarrow \nu/\mu'$$





Time Dilation: the *Work model*

Basic Idea: competing systems in an optimization environment can be simulated simultaneously and get their time scales dilated based on their performance.



Software	Simulated Jobs ['000]	Ratio jobs CSP:TD
Arena	0.325	10:1
PMI	2.6	80:1
Witness	330	10:1
PMII	760	23:1

Optimal Computing Budget Allocation

(A) Determines the relative allocation between non best designs (1), and (B) Derives the relative allocation between the best design and non best designs (2).

We maximize the $PCS = P\left(Y_b < Y_i,\ \forall i \neq b|\mathcal{F}\right)$ (Chen et al. 2000):

$$n_{x_i}/n_{x_j} = \left(\frac{\hat{\sigma}_{x_i}/\delta_{x_b,x_i}}{\hat{\sigma}_{x_j}/\delta_{x_b,x_j}}\right)^2, \quad (1)$$

$$n_{x_b} = \hat{\sigma}_{x_b}\left(x_b\right)\sqrt{\sum_{x\in\mathbb{X}:x\neq x_b}\frac{n_{x_i}^2}{\hat{\sigma}_{x_i}^2}}, \quad (2)$$

Optimal Computing Budget Allocation

(A) Determines the relative allocation between non best designs (1), and (B) Derives the relative allocation between the best design and non best designs (2).
We maximize the $PCS = P\left(Y_b < Y_i,\ \forall i \neq b | \mathcal{F}\right)$ (Chen et al. 2000):

$$n_{x_i}/n_{x_j} = \left(\frac{\hat{\sigma}_{x_i}/\delta_{x_b,x_i}}{\hat{\sigma}_{x_j}/\delta_{x_b,x_j}}\right)^2, \quad (1)$$

$$n_{x_b} = \hat{\sigma}_{x_b}\left(x_b\right)\sqrt{\sum_{x\in\mathbb{X}:x\neq x_b}\frac{n_{x_i}^2}{\hat{\sigma}_{x_i}^2}}, \quad (2)$$

OCBA has been used mainly to allocate *replications*.
We use OCBA *in a time dimension* to assign the time scales $\nu_{\mathbf{x}}\forall \mathbf{x} \in \mathbb{X}^d$.

Proposed Approach

- TD1 Procedures are tailored to the simulation architecture and the way to implement a time dilated experiment.
- TD2 The rules to update the time scales are not standardized and are problem dependent.
- OCBA1 The OCBA needs to be extended to assign time scales.

Proposed Approach

- TD1 Procedures are tailored to the simulation architecture and the way to implement a time dilated experiment.
- TD2 The rules to update the time scales are not standardized and are problem dependent.
- OCBA1 The OCBA needs to be extended to assign time scales.

We Propose:

Proposed Approach

TD1 Procedures are tailored to the simulation architecture and the way to implement a time dilated experiment.

TD2 The rules to update the time scales are not standardized and are problem dependent.

OCBA1 The OCBA needs to be extended to assign time scales.

We Propose:

1. A general rule for updating time scales based on OCBA;

Proposed Approach

TD1 Procedures are tailored to the simulation architecture and the way to implement a time dilated experiment.

TD2 The rules to update the time scales are not standardized and are problem dependent.

OCBA1 The OCBA needs to be extended to assign time scales.

We Propose:

1. A general rule for updating time scales based on OCBA;
2. A general purpose simulation architecture easily adaptable to a time dilated experiment;

Proposed Approach

TD1 Procedures are tailored to the simulation architecture and the way to implement a time dilated experiment.

TD2 The rules to update the time scales are not standardized and are problem dependent.

OCBA1 The OCBA needs to be extended to assign time scales.

We Propose:

1. A general rule for updating time scales based on OCBA;
2. A general purpose simulation architecture easily adaptable to a time dilated experiment;
3. The result is TD–OCBA a single–run simulation optimization framework for stochastic selection.

Time Scale Updating Rule

OCBA ratio calculation - ORC

$$\frac{\alpha_{[i]}}{\alpha_{[j]}} = \left(\frac{\hat{\sigma}_{[i]}/\delta_{b,i}}{\hat{\sigma}_{[j]}/\delta_{b,j}}\right)^2, i,j \in 1,2,...,k, \text{and} i \neq j \neq b,$$
$$\alpha_{[b]} = \hat{\sigma}_{[b]}\sqrt{\sum_{i=1,i\neq b}^{k}\left(\frac{\alpha_{[i]}}{\hat{\sigma}_{[i]}}\right)^2}, \sum_i \alpha_{[i]} = 1, \nu_i = \frac{\alpha_{[b]}}{\alpha_{[i]}}. \quad (3)$$

Standardized time series for variance estimation

$$\hat{\sigma}_i^2 = \frac{1}{B}\sum_{j=1}^{B}\left\{\sum_{g=1}^{m}\left(g\sqrt{\frac{12}{m}}\left(\frac{1}{g}\sum_{l=1}^{g}Y_{i,(j-1)m+l} - \frac{1}{m}\sum_{l=1}^{m}Y_{i,(j-1)m+l}\right)\right)\right\}^2 \quad (4)$$

TD-OCBA Algorithm

Algorithm 1: Time Dilated Optimization & OCBA (TD–OCBA)

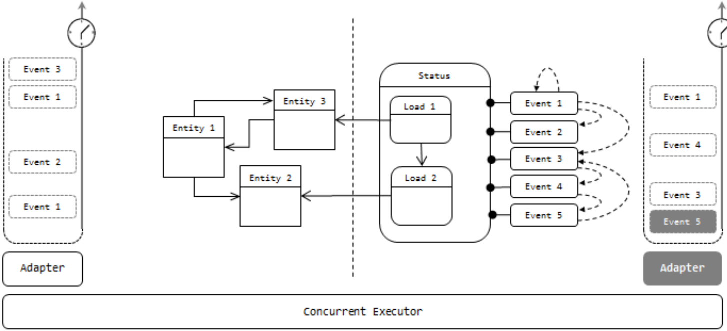
Initialization: Set the total budget $T = \sum_i n_i$ (number of observations);
$l = 0, \nu_{i,k} \leftarrow \nu_0, d_i = 0$;
while $\sum_{i=1}^{k} n_i \leq T$ **do**
  while $\nexists i = 1, \ldots, k \; s.t. j_i = m$ **do**
    Observe $\{Y_{i,d_i m + j_i}\}$ $j_i \leftarrow j_i + 1$;
  end
  for $i = 1, \ldots, k$ **do**
    if $n_i = m$ **then**
      Update: $\bar{Y}_i = \sum_{j=1}^{d_i m + m} \frac{Y_j}{d_i m + m}$, $\sigma_i \leftarrow \sqrt{\mathcal{A}_i}$, $n_i \leftarrow n_i + m$,
      $d_i \leftarrow d_i + 1$;
    end
  end
  Choose b s.t. $b \in \arg\min_{i=1,\ldots,k} \bar{Y}_i$, $v_i \leftarrow \alpha_b / \alpha_i$, $i = 1, \ldots, k$
end

Implementation Details: DES# (Li et al. 2015)

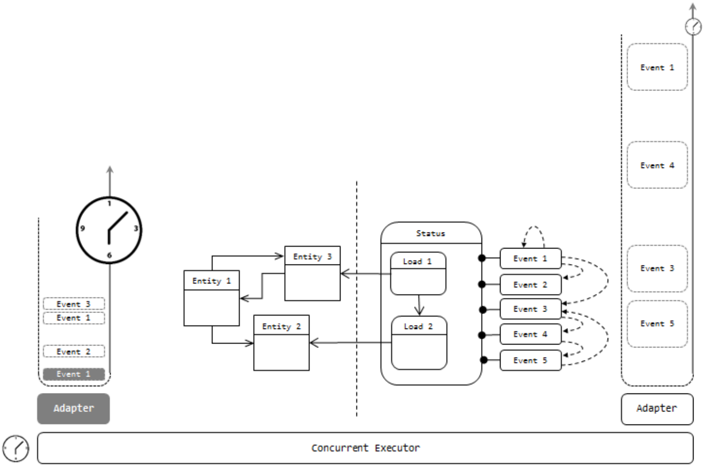
- Each system has its *local event list* associated to the *local clock*.
- At execution the local header event will be *scaled* to the *global clock*.
- The earliest header event in the local list according to the *global clock* is executed.

Due to the C# simulator, we can choose to incorporate the time dilation as a feature available to *all* the models developed within the DE# framework.

Each simulator System clock is synchronized to other simulators through the common clock of the concurrent executor, which selects a model.



Each simulator System clock is synchronized to other simulators through the common clock of the concurrent executor, which selects a model.



Advantages of the Proposed Framework

Achievement

- Any DES model can be easily adapted to run in a time dilated environment.

Advantages of the Proposed Framework

Achievement

- Any DES model can be easily adapted to run in a time dilated environment.
- Time scale update rule is not problem specific.

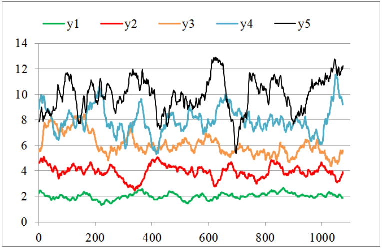
Experimental Settings

Two groups of numerical experiments are employed to test the performance of TD-OCBA and compare it with original TD.

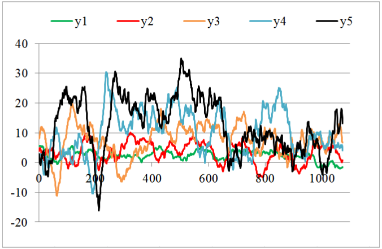
1. Time series tests
   - Objective: identify the smallest asymptotic mean;
   - We show the convergence of the algorithm and effects of the batch size.
2. A job–shop systems
   - Finite set of configurations each increasing one machine type.
   - We compare TD–OCBA with the TD implementation in Hyden et al. (2000).

Simulation Results: Time series tests

$$Y_{i,t+1} = c_i + \phi Y_{i,t} + \epsilon_i,\ c_i = i,\ \phi = 0.5\ i = 1, ..., 5$$
$$(1)\epsilon_i \sim \mathcal{N}(0, \delta \cdot i), (2)\epsilon_i \sim \mathcal{U}(-\sqrt{3}\delta \cdot i, \sqrt{3}\delta \cdot i).$$
$$\mu_i = \frac{c_i}{1-\phi} = 2i,\ i = 1, \ldots, 5.$$

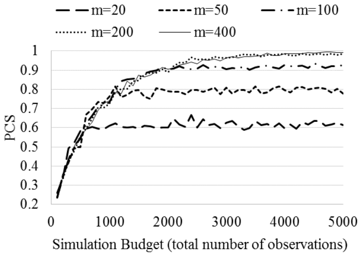


(a) $\delta = 1.0$

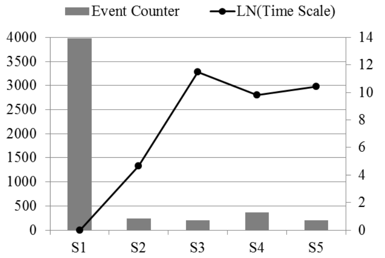


(b) $\delta = 6.0$

Simulation Results: Time series tests



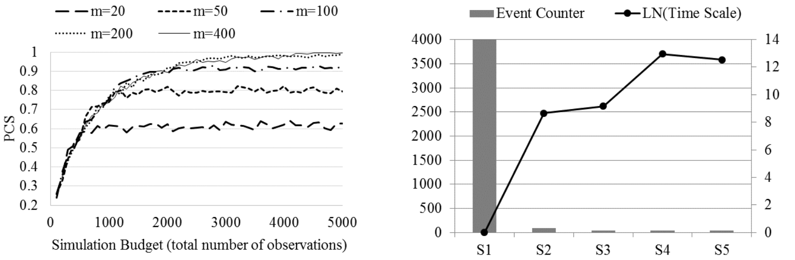
(a) Impact of the batch size *m* over the PCS



(b) Final allocated computing effort and time scales

Figure 1: Results from experiment (1) with normally distributed noise, $\delta = 6.0$

Simulation Results: Time series tests



(a) Impact of the batch size m over the PCS

(b) Final allocated computing effort and time scales

Figure 2: Results from experiment (1) with uniformly distributed noise, $\delta = 6.0$

Job–shop Problem

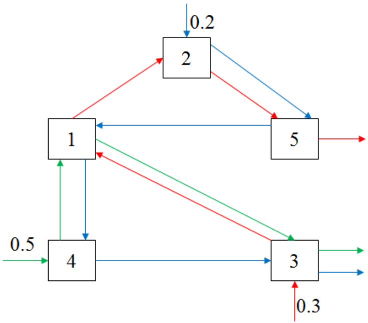


Figure 3: Job Shop System

Candidates Definition:

- 5 machine types $\boldsymbol{n} = \{6, 5, 7, 6, 4\}$. 5 candidates: $\{n_1 + 1, \ldots, n_5 + 1\}$
- System Time $\vartheta_i = \{13.17, 15.56, 15.98, \mathbf{11.28}, 15.79\}$;
- Configuration $i = 4$ is the best.

Benchmarking:

- $m = 100$, warm–up period of 2000[jobs];
- In Hyden et al. [2000] (TD approach) the time scales are updated according to $V_i / D_i^2, \; i = 1, \ldots, 5$;

Simulation Results: Job–shop Problem

All the algorithms tested use time dilation:

Total budget T	50000	100000	200000	500000
PCS(EA)	0.5937	0.6826	0.7669	0.8247
PCS(TD)	0.6609	0.7701	0.8704	0.9585
PCS(TD–OCBA)	0.6710	0.7821	0.8795	0.9660

Table 1: Table of PCS using TD and TD–OCBA

Table 1 shows the outcomes from 10000 replications, suggesting TD-OCBA performs consistently better than TD and both outperform Equal Allocation (EA).

Conclusions

Achievements

1. Time Dilation was enhanced by integrating it with the OCBA;
2. OCBA was adopted to stochastically and dynamically set the simulation parameters;
3. Two groups of experiments illustrate the performance of TD–OCBA.

Futures

1. The importance of the batch size was observed and the dynamic batch sizing is under investigation;
2. A new OCBA rule for time series observations, which should improve the performance of TD–OCBA when the available total budget T is particularly small, is being tested.

Thank You

Dr. Giulia Pedrielli,
cmsgp@nus.edu.sg