MockSAS: Facilitating the Evaluation of MAB in SAS

Elvin Alberts
Vrije Universiteit Amsterdam

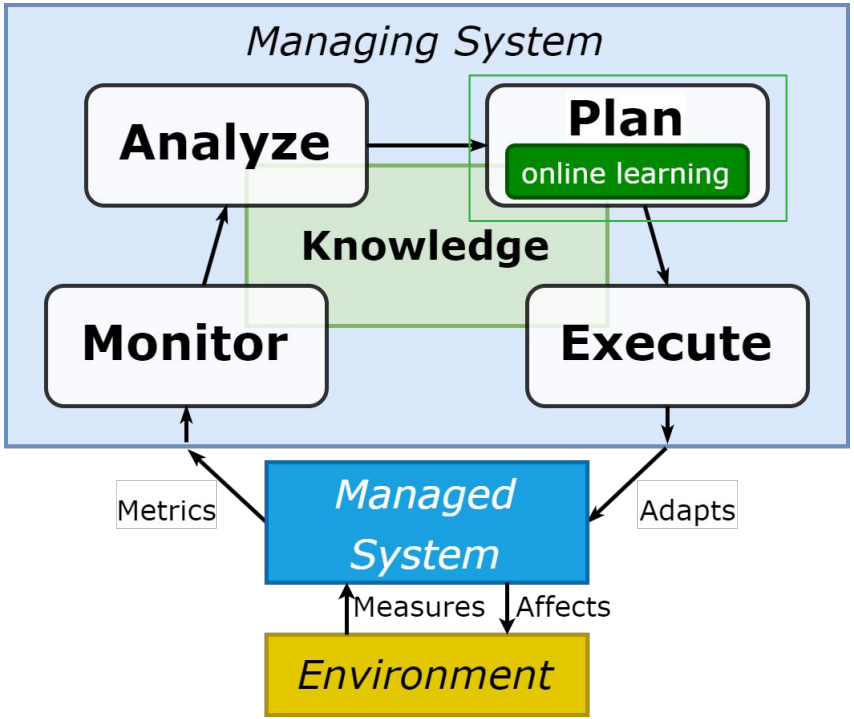
Ilias Gerostathopoulos
Vrije Universiteit Amsterdam

Tomas Bures
Charles University in Prague

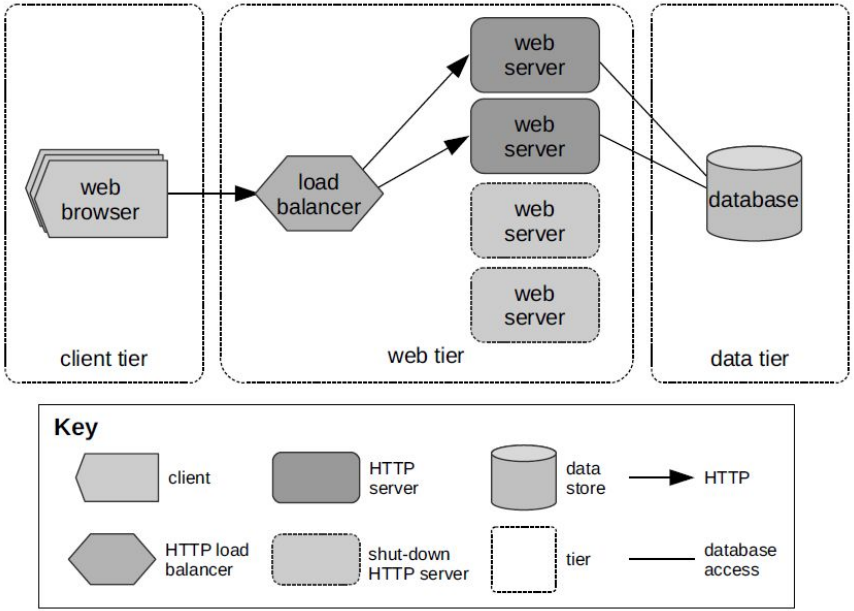
SOFTWARE AND SUSTAINABILITY
S2 RESEARCH GROUP

VU VRIJE UNIVERSITEIT AMSTERDAM

Self-Adaptive Systems and Online Learning



SWIM: Simulator for Web Infrastructure and Management[1]



[1]Moreno, Gabriel A., Bradley Schmerl, and David Garlan. "Swim: an exemplar for evaluation and comparison of self-adaptation approaches for web applications." *2018 SEAMS*. IEEE, 2018.

Multi-armed Bandits

Arms:

$$A = \{a_1, a_{...}, a_k\}$$

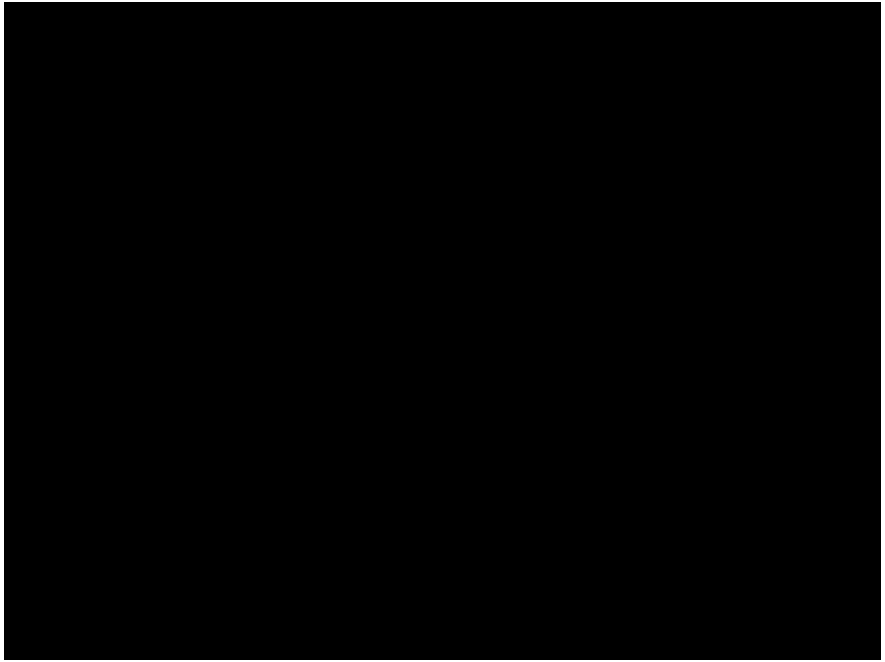
Horizon: *n*, total number of rounds

Context:

$$C_{t \in [n]} = \{R(a_1), R(a_{...}), R(a_k)\}$$

Regret:

$$\mu^* n - \mu_j \sum_{j=1}^{K} \mathbb{E}[T_j(n)] \quad \text{where } \mu^* = \max_{i=1,\dots,K} \mu_i$$



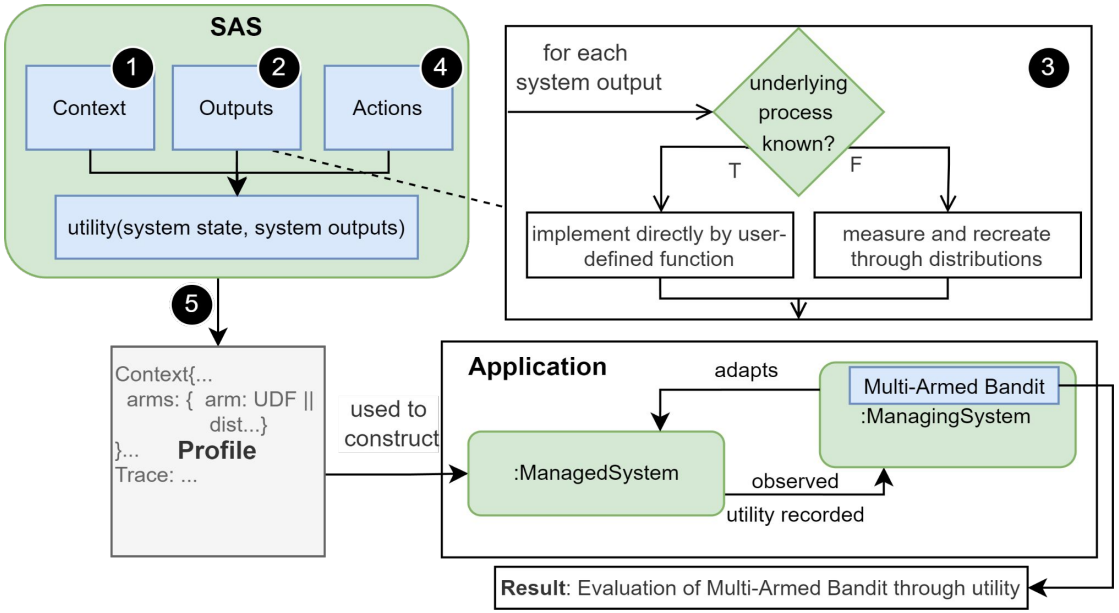
UCB Tuned

Problems in Evaluating MAB Policies

- Operating environments are unpredictable, and may contain unknown unknowns
- Statistically significant evaluations cost a lot of time when run on ‘real’ systems
- Reproducing system states and extrapolating new ones is tedious manual work

A reusable and extensible method for more time-efficient evaluation of MAB policies with SAS

Approach: MockSAS



1. Determine the system context(s) to be profiled.
2. Identify which outputs influence the calculation of the reward.
3. Of the identified outputs, determine which require profiling.
4. Identify the arms (adaptation actions) per context whose effects on the reward should be measured.
5. For each arm in each context, collect sufficient values of the profiled outputs to be able to identify their distribution.

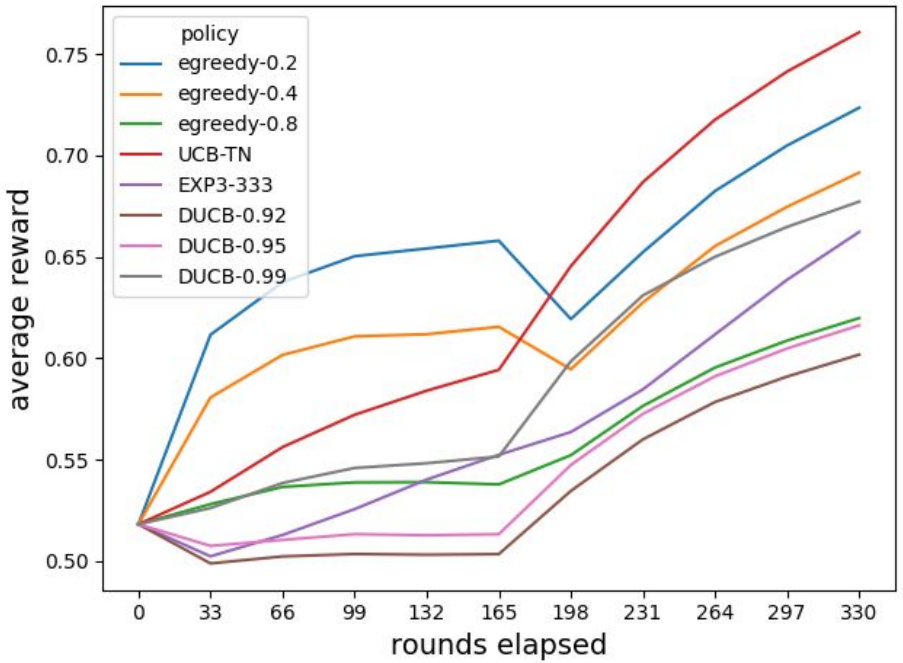
Example

```
60Requests {
   features: { rq_rate: uniform(54,66)}

   arms: {
      3Servers: utilitySWIM(rq_rate, normal(.064,.009)…)
      6Servers: utilitySWIM(rq_rate, normal(.039,.001)…)
      8Servers: utilitySWIM(rq_rate, normal(.039,.001)…)
      …
   }}
80Requests {
   features: { rq_rate: uniform(72,88) }

   arms: {
      3Servers: utilitySWIM(rq_rate, normal(1.68,1.44)…)
      6Servers: utilitySWIM(rq_rate, normal(.041,.001)…)
      8Servers: utilitySWIM(rq_rate, normal(.039,.001)…)
      …
   }}
Trace: (60Requests, 166) (80Requests,166)
```

(Subset of) Results



Policy Name	Mean Reward	Median Reward	Average Ranking	Percentage Matching SWIM
UCB-TN	*0.77*	*0.76*	*1.13*	*86.67%*
egreedy-0.2	0.73	0.73	2.03	76.67%
egreedy-0.4	0.69	0.69	3.1	73.33%
DUCB-0.99	0.68	0.68	4.07	23.33%
EXP3-333	0.66	0.66	4.7	26.67%
egreedy-0.8	0.62	0.62	6.47	56.67%
DUCB-0.95	0.62	0.62	6.6	60.00%
DUCB-0.92	0.6	0.6	7.9	0.00%

Looking Ahead

- Comparing more online learning solutions across paradigms (not just reinforcement learning)
- Applying MockSAS to more SAS.
- Automating the profiling process further through distribution detection etc.

Code can be found at:

https://github.com/EGAlberts/MockSAS
https://github.com/EGAlberts/MASCed_bandits
https://github.com/EGAlberts/swim