Machine learning for caching systems

Online Learning for Caching with Heterogeneous miss-costs

Robert Vadastreanu

Responsible professor: Dr. Georgios Iosifidis

Supervisors: Naram Mhaisen, Fatih Aslan

Introduction

- Improving efficiency of a system by keeping frequently requested items easily accessible
- Small memory with high speed, closed to the CPU
- The classical policies: LRU, LFU or FIFO
- Introducing two new algorithms Universally Adaptive Caching Algorithm (UAC) and Adaptive Per File Caching Algorithm (APFC)

Background

- OGA outperforms LFU and LRU by 20% in some scenarios [1]
- Online learning with adaptive steps is more efficient than learning with fixed steps [2]

System model

A request is chosen from the set:

$$X = \{\boldsymbol{x} \in \{0,1\}^N \mid \sum_{i=1}^{N} x^i = 1\}$$

A caching config is chosen from:

$$\mathcal{Y} = \left\{\boldsymbol{y} \in [0,1]^N \mid \sum_{i=1}^{N} y^i \leq C\right\}$$

The utility function is defined as:

$$f_t(\boldsymbol{x_t}, \boldsymbol{y_t}) = \sum_{i=1}^{N} w_t^i x_t^i y_t^i$$

$$\boldsymbol{w_t} = \left(w_t^i \middle| i \in \{1\ 2, .. \mathrm{N}\}, 0 < w_t^i\right)$$

Problem statement

Simulation for horizon T:

$$\boldsymbol{X} = (\boldsymbol{x_1}, \boldsymbol{x_2}, \dots \boldsymbol{x_T}) \quad \boldsymbol{Y} = (\boldsymbol{y_1}, \boldsymbol{y_2}, \dots \boldsymbol{y_T})$$

The regret is defined as:

$$R_T(\boldsymbol{X}, \boldsymbol{Y}) = U(\boldsymbol{X}, \boldsymbol{Y}^*) - U(\boldsymbol{X}, \boldsymbol{Y})$$

Weighted request: $\boldsymbol{g_t} = \boldsymbol{w_t} \odot \boldsymbol{x_t}$

OGA [1] $\boldsymbol{y_{t+1}} = \Pi_Y(\boldsymbol{y_t} + \eta_t \nabla \boldsymbol{f_t})$

- $\Pi_{\mathcal{Y}}(\boldsymbol{z}) = \arg\max_{\boldsymbol{y} \in \mathcal{Y}} ||\boldsymbol{z} - \boldsymbol{y}||$
- $\nabla \boldsymbol{f_t} = \left(\frac{\partial f_t}{\partial y_t^i}, \forall i \in \{1,2, \dots N\}\right)$
- L – upper bound for $||\nabla \boldsymbol{f_t}||$
- $\eta_t = \frac{\mathrm{diam}(\mathcal{Y})}{L\sqrt{T}}, \forall t \in \{1,2, \dots T\}$
- $R_T(\boldsymbol{OGA}) \leq diam(\mathcal{Y}) L \sqrt{T}$

Adaptive caching

UAC (Universally Adaptive Caching)

$$\eta_t = \frac{\sqrt{2} diam(\mathcal{Y})}{2\sqrt{\sum_{j=1}^{t} ||\boldsymbol{g_i}||^2}} \; [3]$$

$$R_T(\boldsymbol{UAC}) \leq \sqrt{2} diam(\mathcal{Y}) \sqrt{\sum_{t=1}^{T} ||\boldsymbol{g_t}||^2}$$

APFC (Adaptive Per File Caching)

$$\eta_t^i = \frac{\sqrt{2} diam(\mathcal{Y})}{2\sqrt{\sum_{j=1}^{t} (w_j^i x_j^i)^2}} \; [2]$$

$$R_T(\boldsymbol{APFC}) \leq \sqrt{2} \sum_{i=1}^{N} \sqrt{\sum_{j=1}^{T} (w_j^i x_j^i)^2}$$

Results

Figure 1 - Unweighted MovieLens Simulation

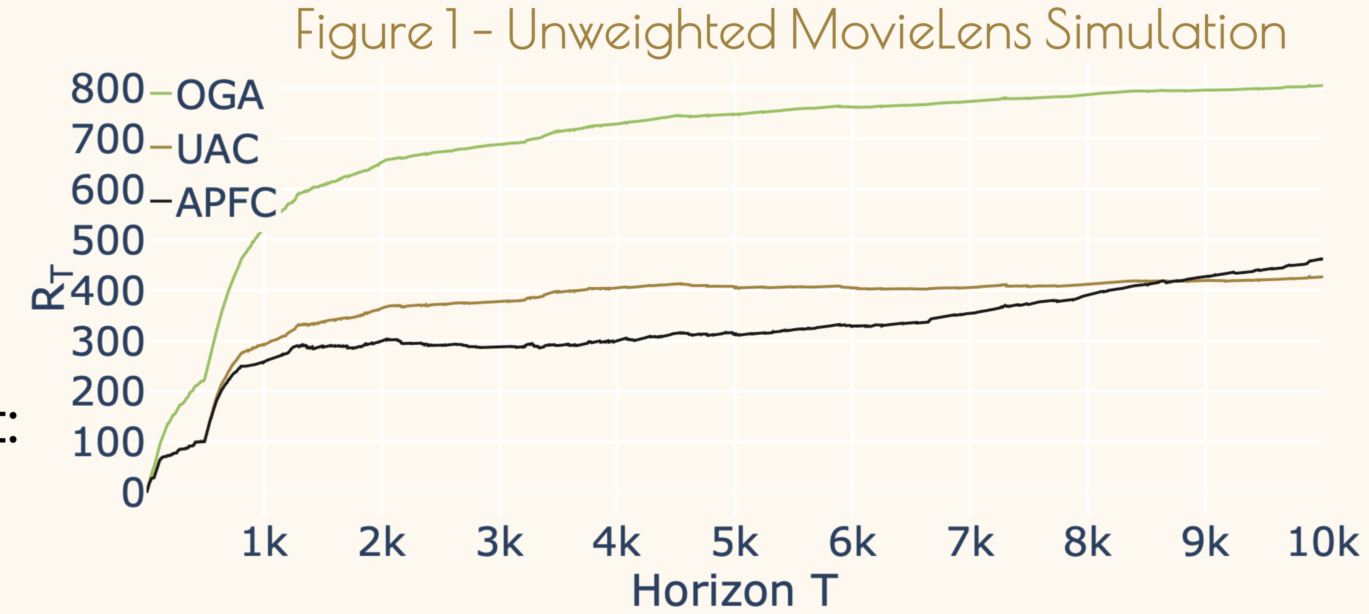


Figure 2 - Weighted MovieLens Simulation

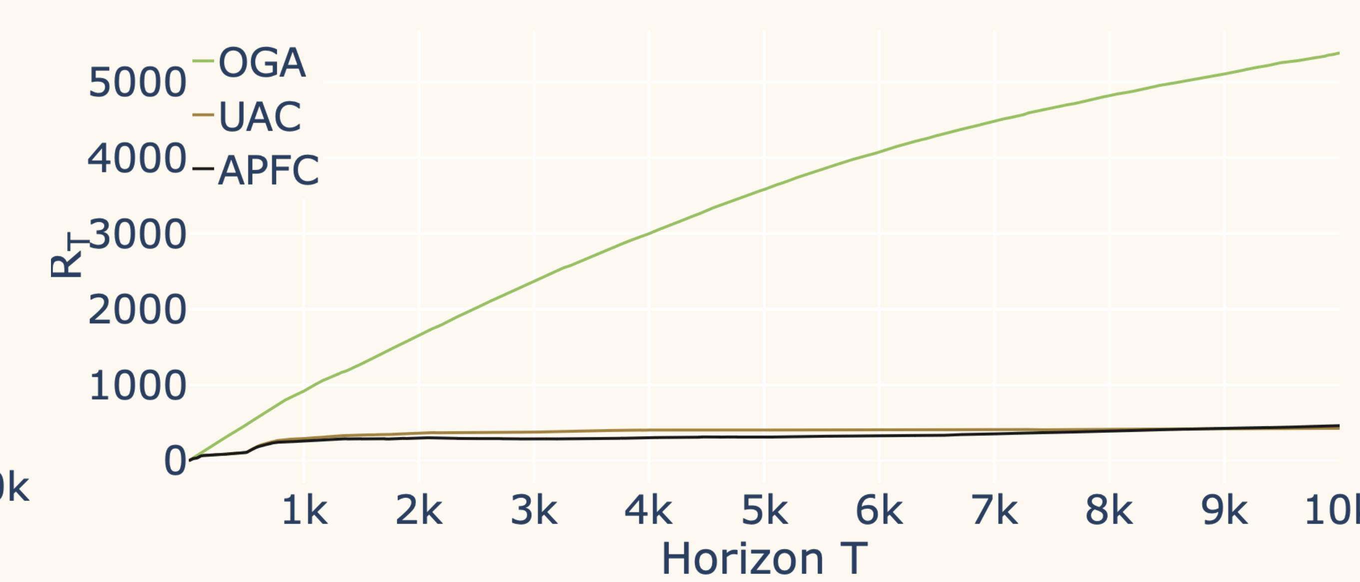


Figure 3: 100 trends simulation

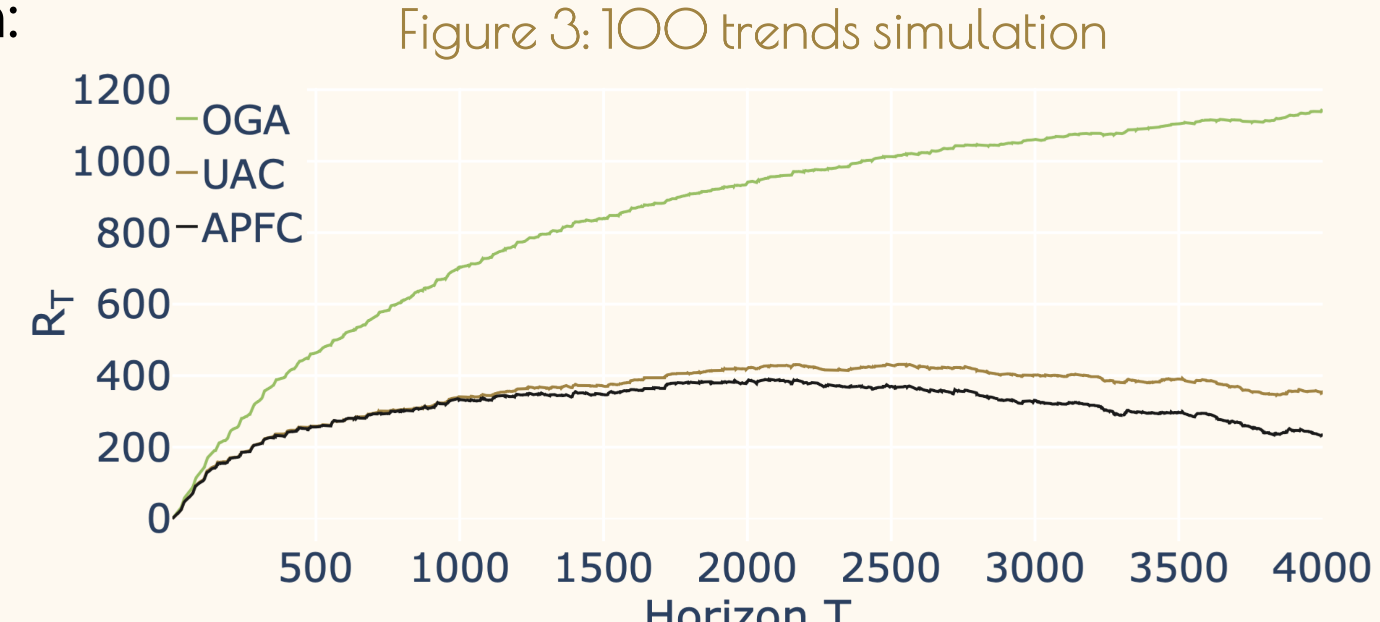
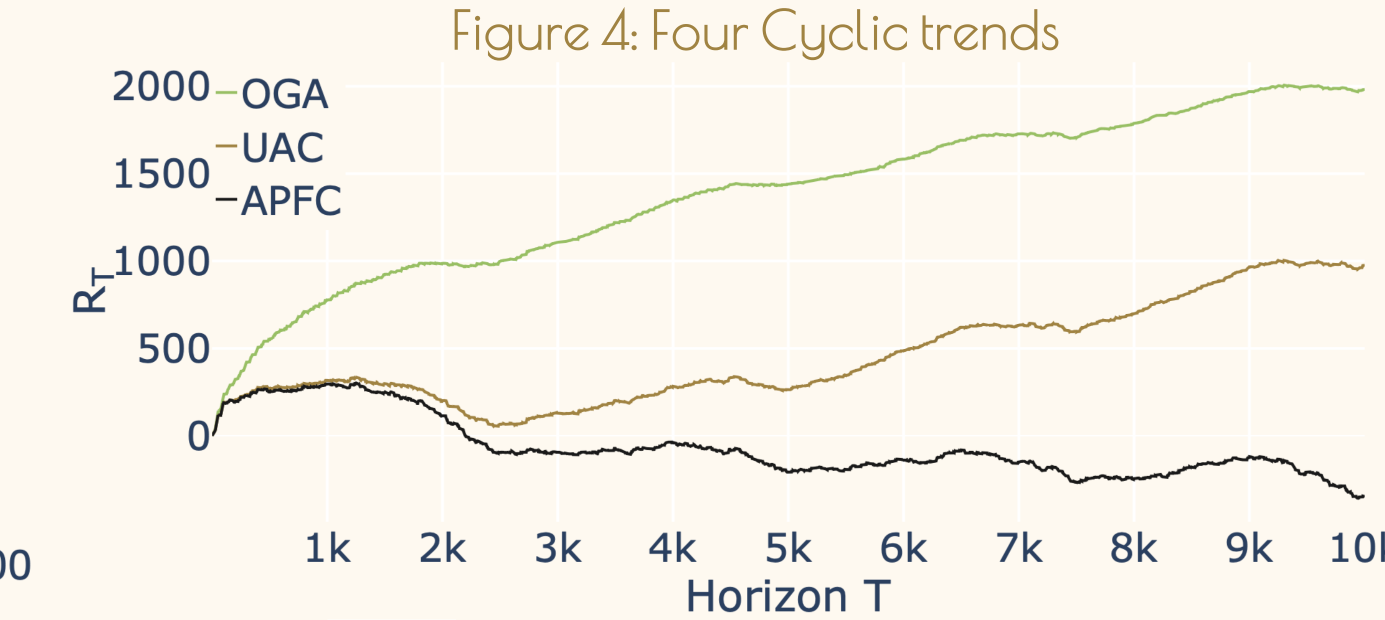


Figure 4: Four Cyclic trends



Limitations

- Online learning is pessimistic
- Requires time to compute regret and next configuration

Conclusion

- UAC and APFC adapts well when files' weights are changing
- APFC performs well with trends
- Both algorithms outperformed OGA which outperform the classical methods
- Improvement in caching algorithms

References

[1] L. Vigneri G. Iosifidis G. Paschos, A. Destounis. "Learning to Cache with No Regrets". In IEEE INFOCOM, 2019

[2] *F. Orabona, "A Modern Introduction to Online Learning," arXiv:1912.13213, 2023*

[3] H. B. McMahan, "A Survey of Algorithms and Analysis for Adaptive Online Learning," J. Mach. Learn. Res, 2017.

Contact

Robert Vadastreanu

R.V.Vadastreanu-1@student.tudelft.nl

TU Delft