To send or not to send: Reducing the cost of data transmission

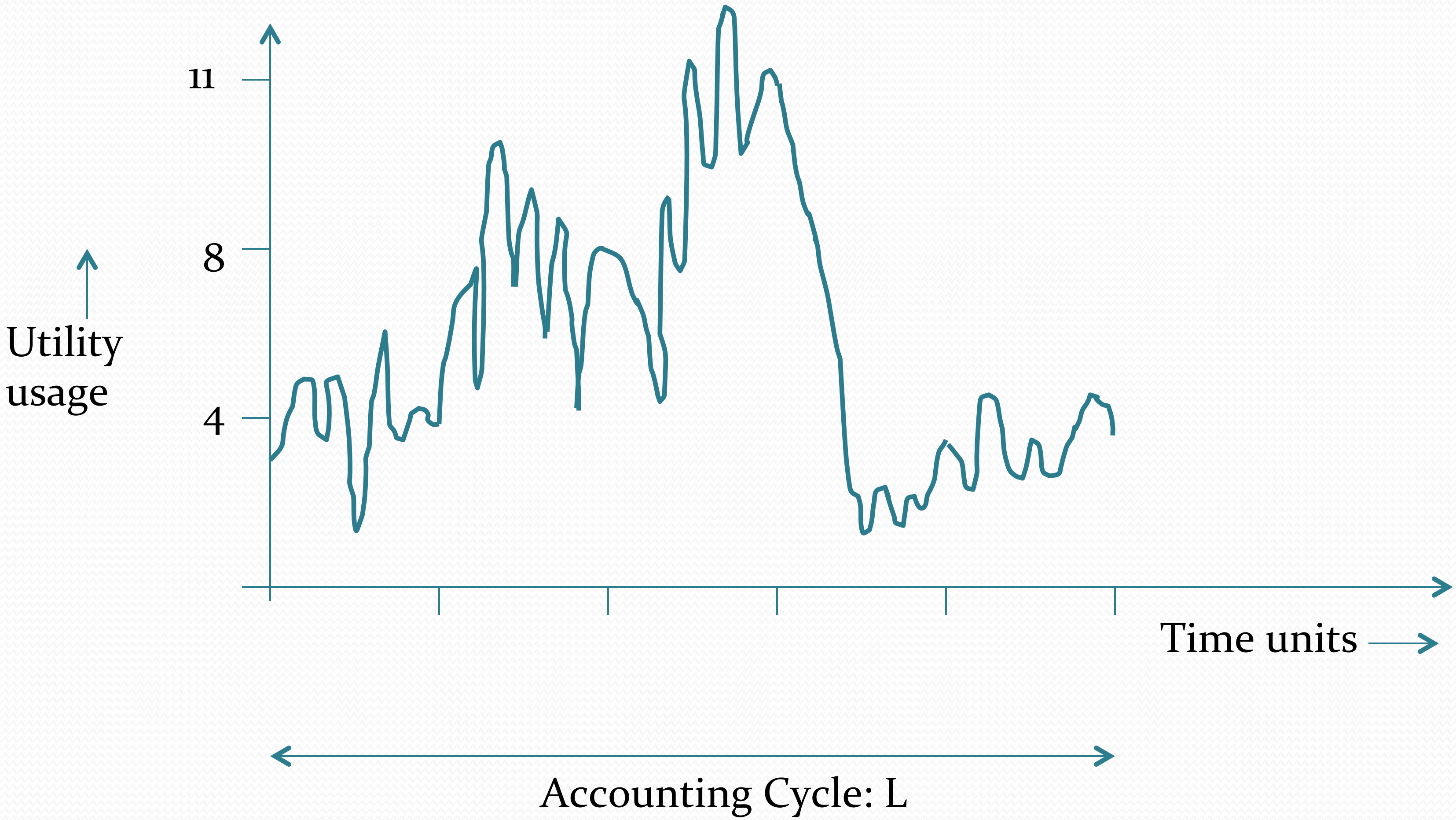
Leana Golubchik, Samir Khuller, Koyel Mukherjee, Yuan Yao

INFOCOM 2013

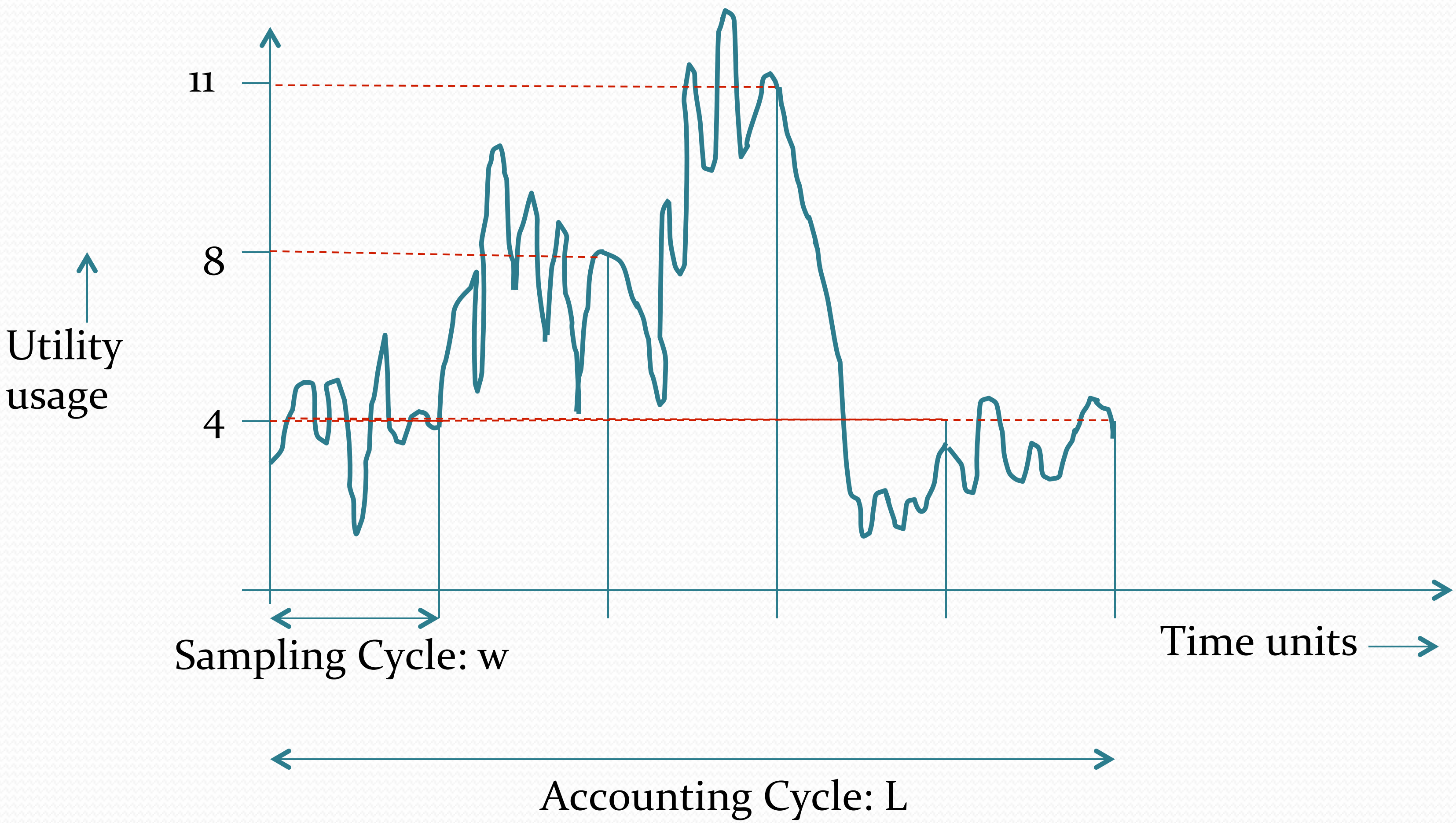
Network Bandwidth- A Costly Resource

- Large volume of data transmitted daily over high speed links.
- High associated costs for network bandwidth.
- Cost accounting performed differently from other utilities.
- Billing is not by the total volume of utility consumed.
- ISP's use the 95^{th} percentile widely for charging.

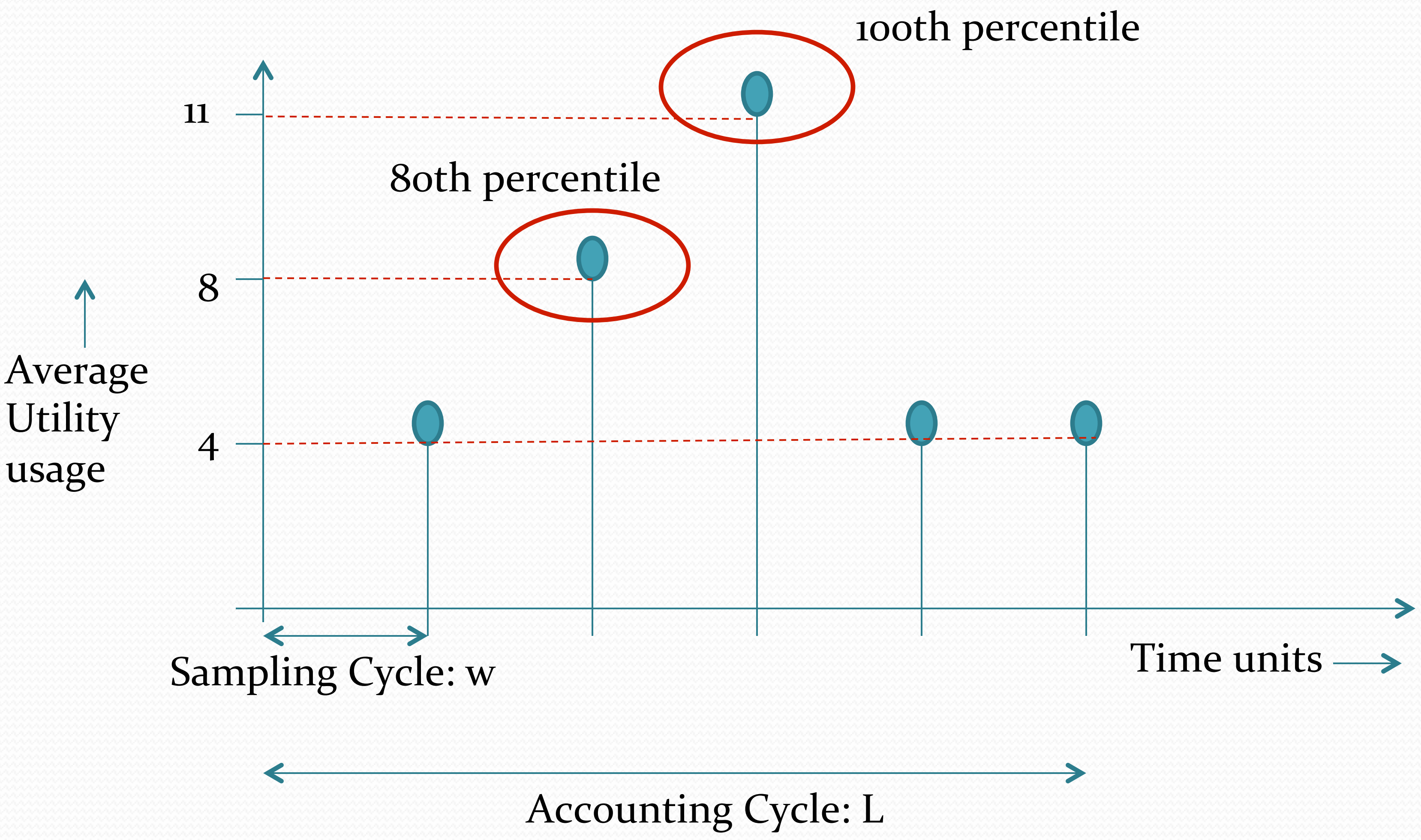
Network Traffic (UMD for example)



Network Traffic

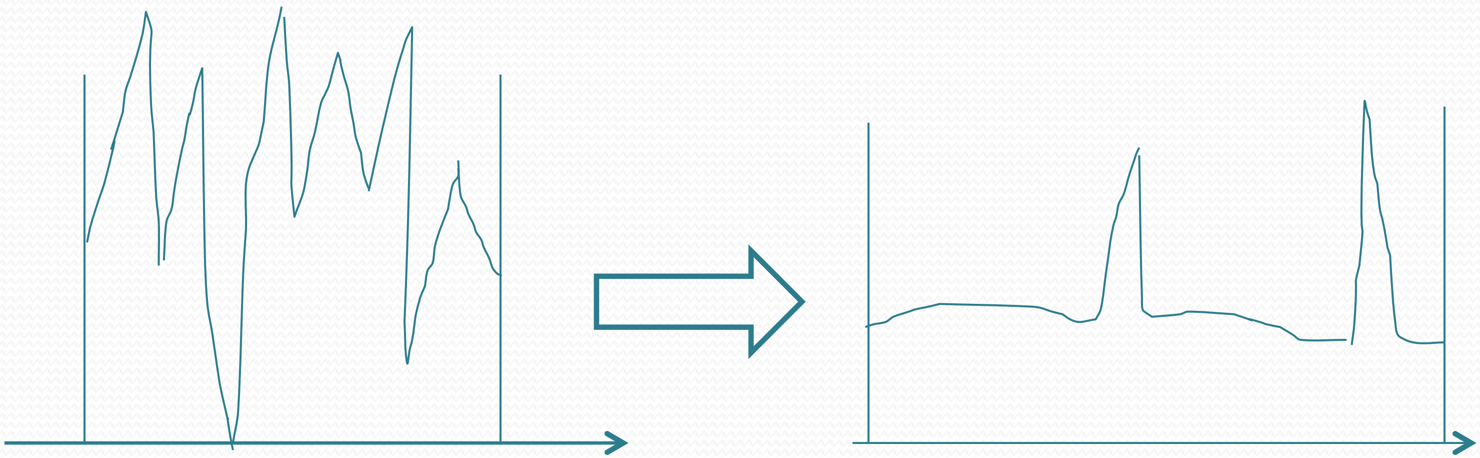


The Percentile Rule



Why is ‘Percentile’ a good measure?

- May help to smoothen the network traffic…
- Potentially avoid network congestion…
- Allow ISP’s to distribute network resources among several organizations at the same time…

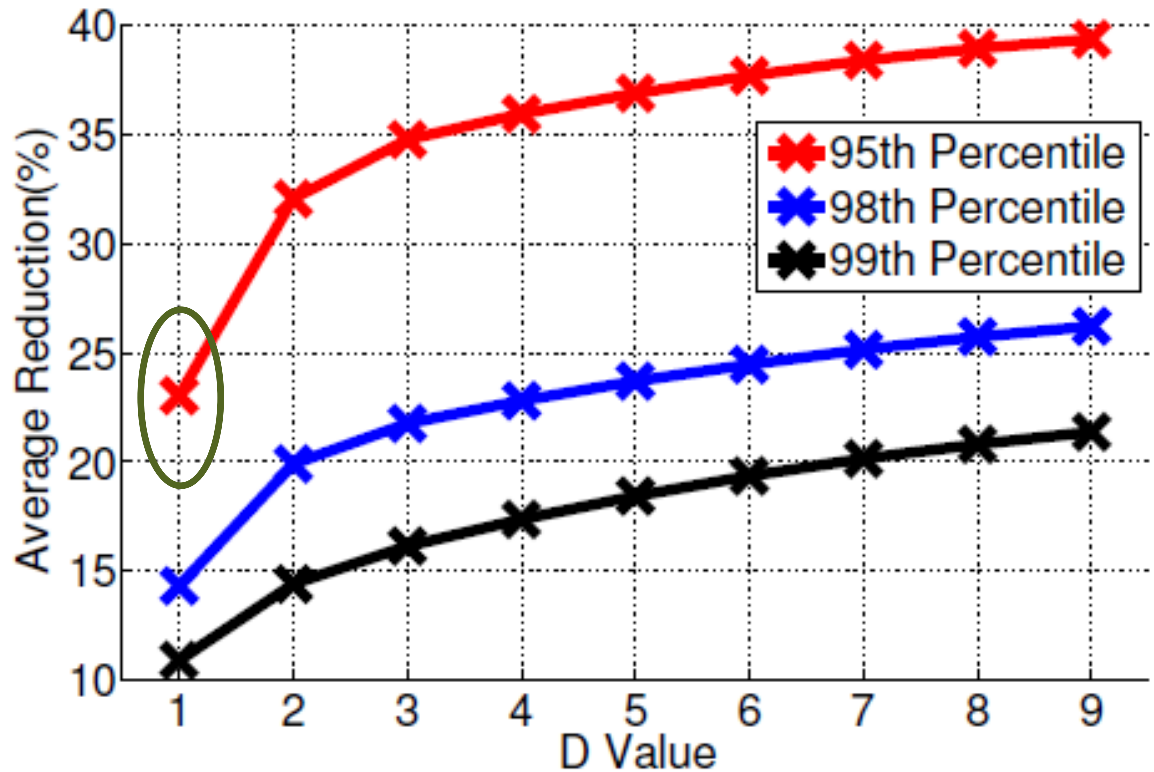


Optimization Opportunity

- The performance requirements of such data transfers include transmission delay:
  - Data chunks have to be delivered within some time period.
  - Not necessary to transmit it at arrival

- Opportunity to optimize data transfers costs!

Possible Cost Savings when Delay is Allowed



Data set: 6 times series generated with different characteristics using the Wikipedia hit records as the basis.

Scheduling Problem

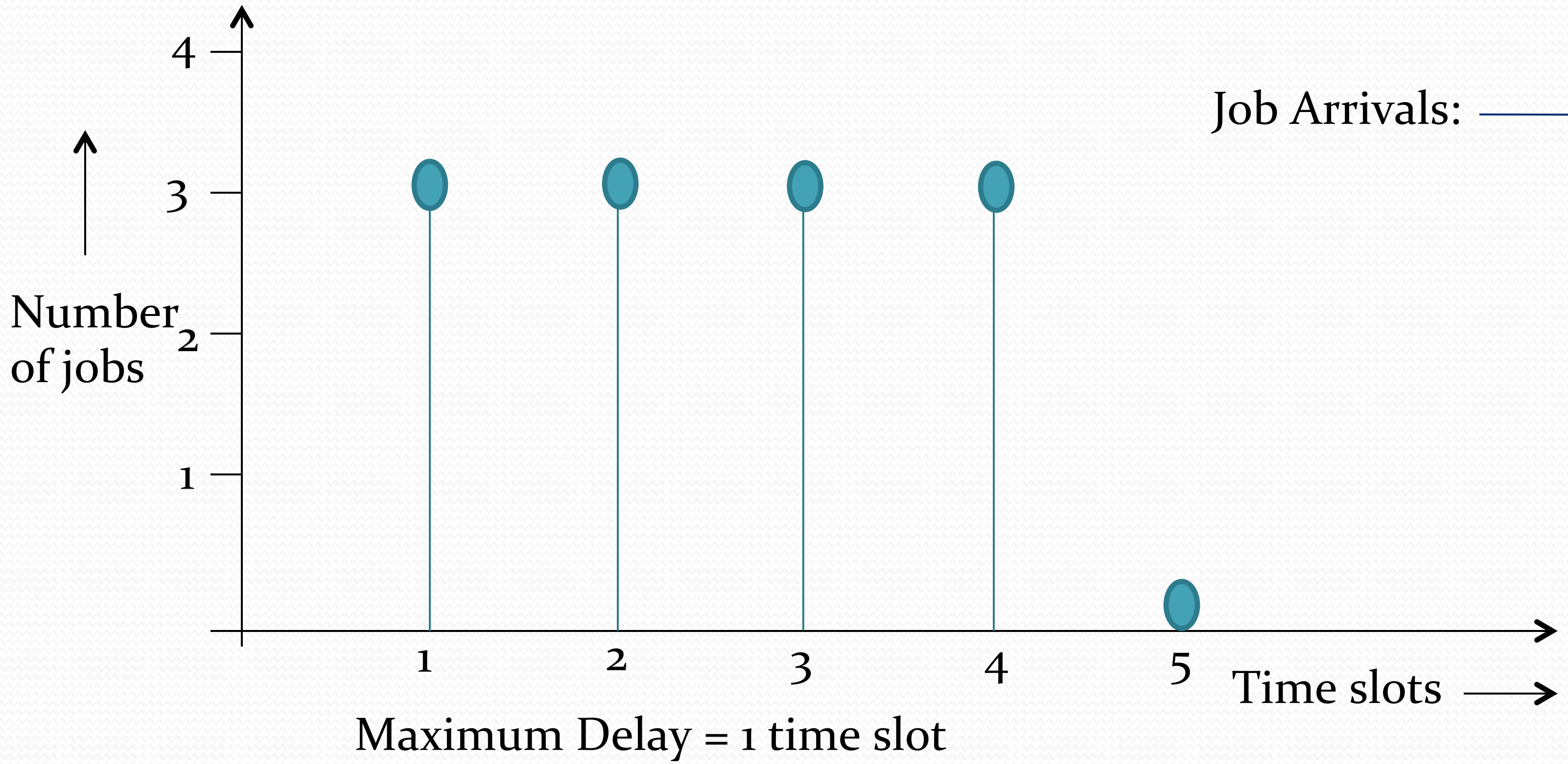
- Data chunks $\Longrightarrow$ unit jobs
- Links (senders) $\Longrightarrow$ servers
- Jobs can be delayed for D time units.
- All jobs need to be done.
- $A(i)$: number of jobs arriving in time slot $i = 1, 2, \ldots, N$.
- $S(i)$: number of jobs served in time slot $i = 1, 2, \ldots, N$.
- Servers have capacity C.

Goal: Choose $S(i)$ to minimize 95$^{\text{th}}$ percentile.

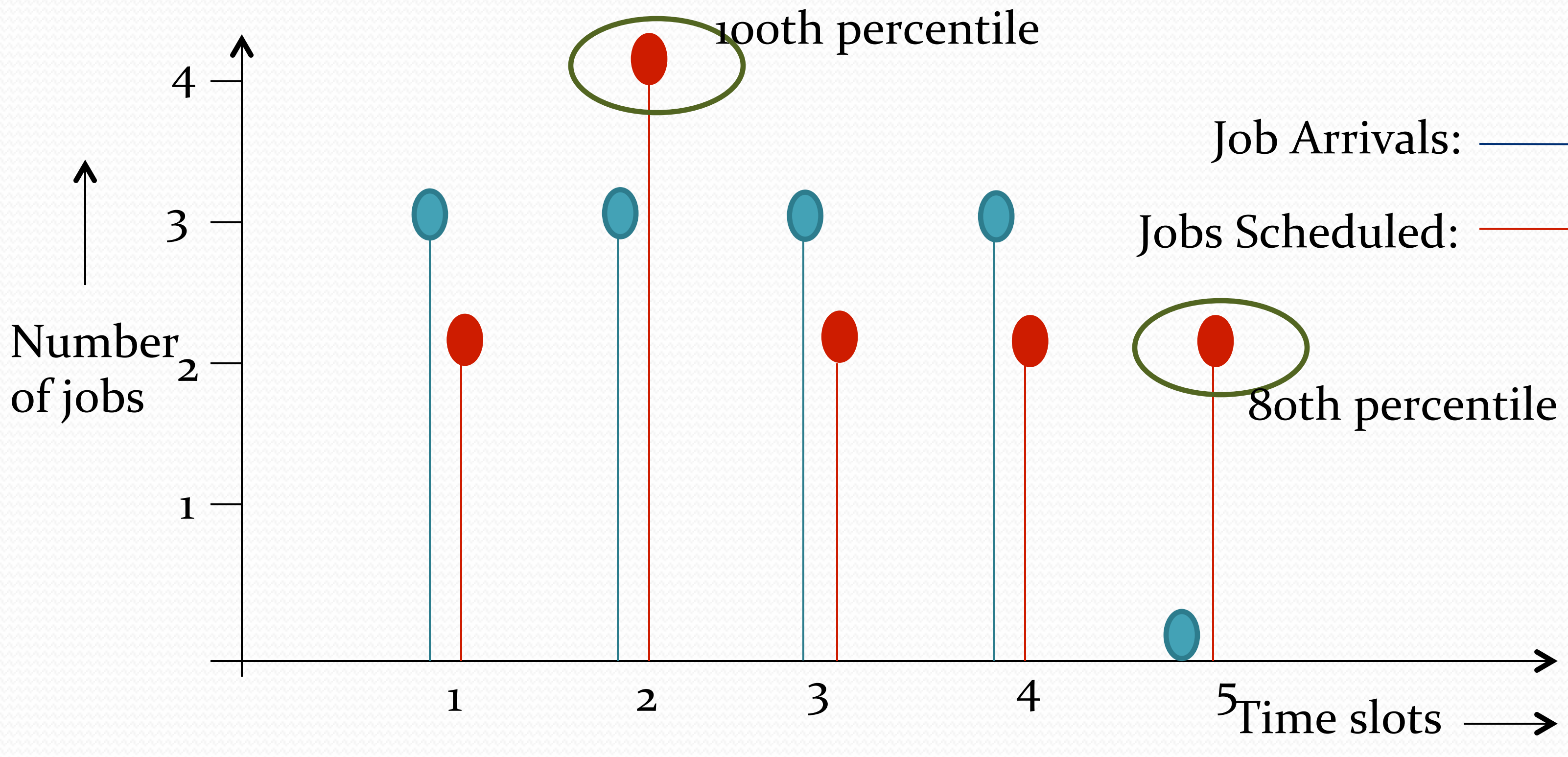
Related Work

- Bounded delay buffer management problem, introduced by Kesselman et al. (STOC 2001).
  - Andelman et al. (SODA 2003)
  - Zhu (J. Algs, 2004)
  - Bartal et al. (STACS 2004, JODA 2006)
  - Li et al. (SODA 2005, 2007)
  - Englert and Westermann (SODA 2007),
  - Bienkowski et al. (SODA 2009)
  - Jez (STACS 2010, ESA 2011)

An Example


4
3
2
1
Number of jobs
Job Arrivals:
1
2
3
4
5
Time slots
Maximum Delay = 1 time slot

Example (contd.)



Maximum Delay = 1 time slot

Our Results

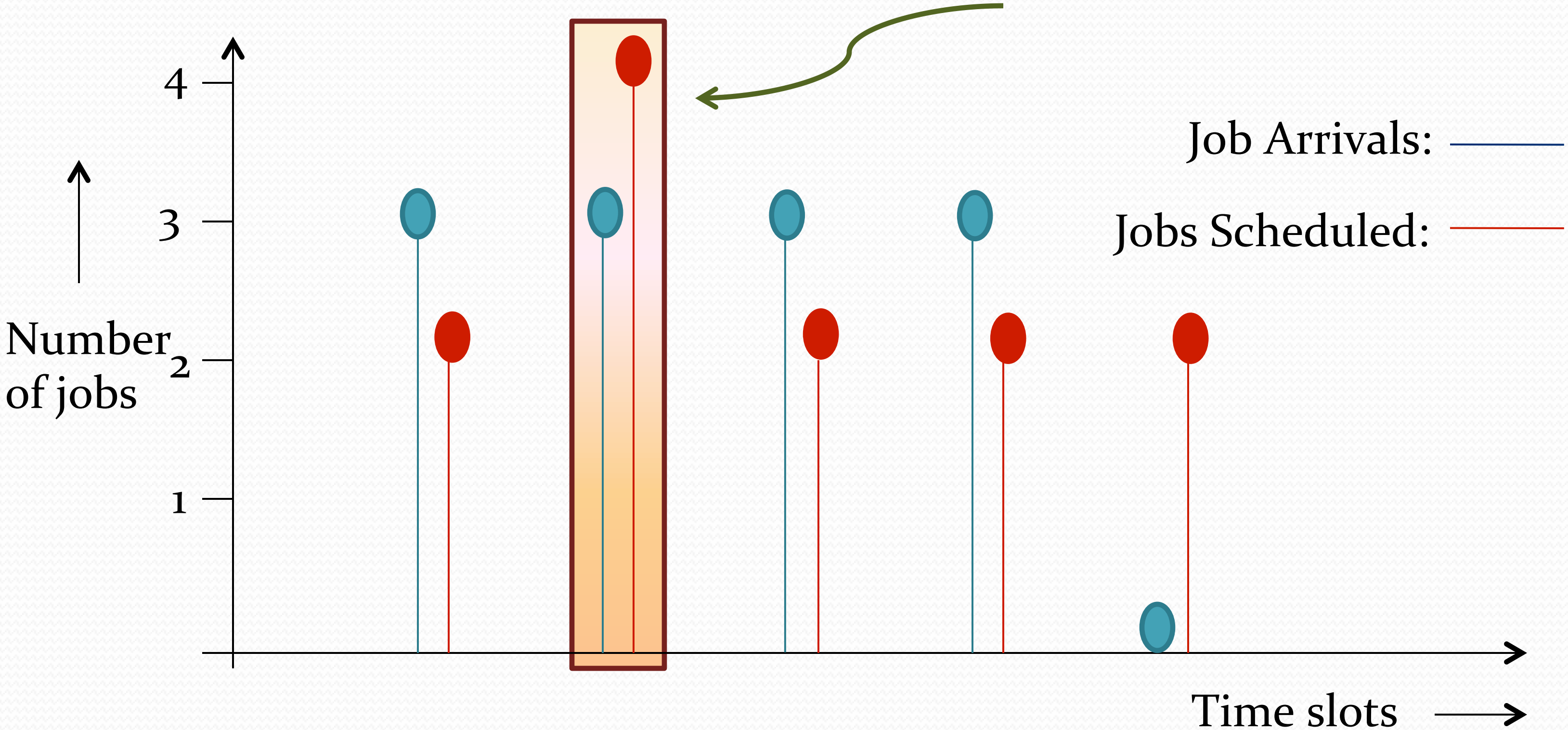
- Optimal offline algorithm for minimizing any specified percentile where jobs can be delayed by D time slots.
- No bounded competitive ratio possible in the online setting.
- Online algorithm for the min-max problem.

  (The min-max problem is similar to the 100^{th} percentile problem.)
- Lower bound on the performance of any online algorithm for min-max problem, almost matching upper bound.
- Propose and test an online heuristic for the percentile problem.

Offline Problem for D=1

Offline Problem for D=1

- Job arrivals $A(i), i = 1, 2, ..., N$ known a priori.
- Jobs served in the top 5% of the total time slots are not included in accounting. Consider these slots "free".
- Let $T = 5\% \times N$ be the number of free time slots



Offline Problem for D=1 (contd.)

- Free slots: up to C jobs.
- Accounted $N-T$ slots, upto H jobs ($H \le C$).
- Find the smallest feasible H.
- Guess H, verify if feasible. Use binary search for H.

Source: beyondinsurance.wordpress.com

Offline Problem for D=1 (contd.)

- Define $OPT(i,t)$: the minimum number of total jobs left unfinished by the end of time slot i, using t "free" slots.

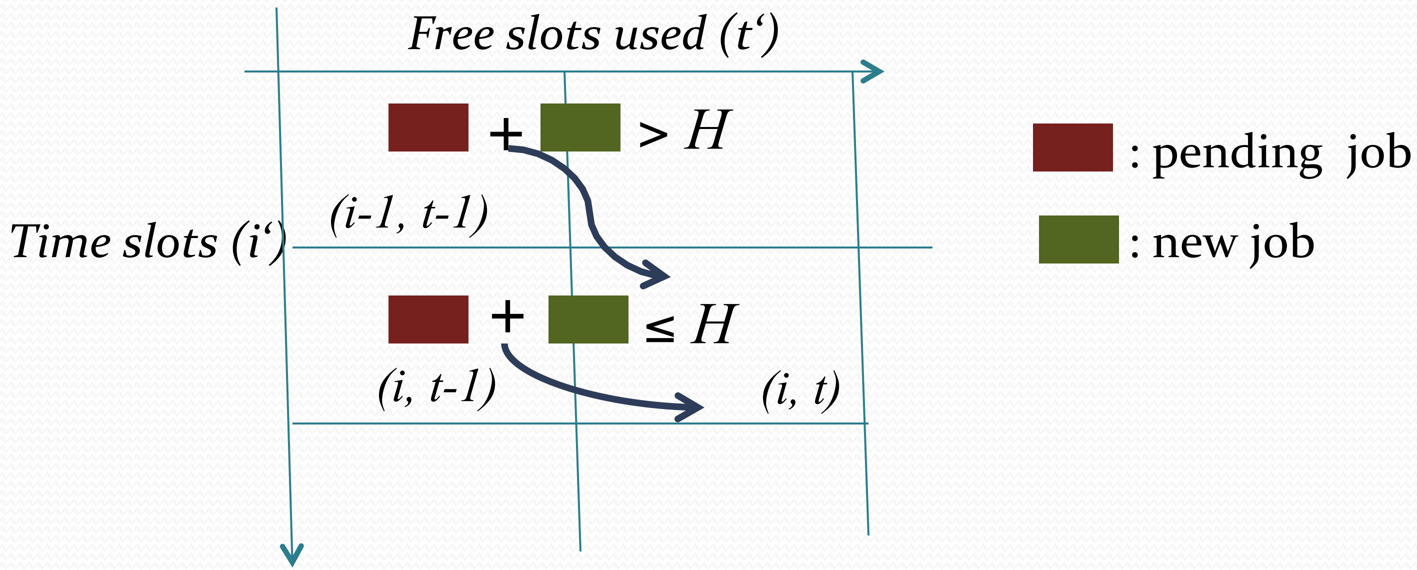
$$OPT(1,0) = \max\{A(1) - H, 0\}$$

$$OPT(1,1) = 0$$

$$OPT(i,t) =$$

$$\min \begin{cases} \max\{OPT(i-1,t-1) + A(i) - C, 0\} \\ \max\{OPT(i-1,t) + A(i) - H, 0\} \text{ if } OPT(i-1,t) \leq H \end{cases}$$

Offline Problem for D=1 (contd.)



Theorem 1. $OPT(i,t)$ minimizes the number of unserved jobs in the first i slots, assuming that at most t slots can have load $> H$, and all the other slots have load at most H.

Offline Problem for D=1 (contd.)

- Proof by induction.
- Assume claim is true for $(i-1, j)\ \forall\ j \le (i-1)$.
- If in optimal solution, slot i has load $> H$, then we can use upto capacity C here.
- Else, if opt has load $\le H$ in slot i, and if more than H jobs are passed on, then it is not feasible since $D = 1$.

H feasible $\Longleftrightarrow$ $OPT(N,T) = 0$

Offline Problem for D>1: General Uniform Delay

General Uniform Delay D

- $D > 1 \Rightarrow$ Jobs can be delayed for 1, 2, ..., D time slots.

- Service priority should be different: Earliest first

- Let $\mathbf{U(i)} = (U_1(i), \ldots, U_D(i))$ be the vector of unfinished jobs at the end of time slot i.

- $U_D(i)$: Jobs unserved for D time slots.

General Uniform Delay D (contd.)

Dynamic program to calculate $OPT(i,t)$.

$$OPT(1,0) = \max(A(1) - H, 0)$$

$$OPT(1,1) = 0$$

$$OPT(i,t) = 0 \ \forall \ t \geq i$$

$$OPT(i,t) =$$

$$\min \begin{cases} \max\{OPT(i-1,t-1) + A(i) - C, 0\} \\ \max\{OPT(i-1,t) + A(i) - H, 0\} \text{ if } U_D(i-1,t) \leq H \end{cases}$$

Theorem 2. $OPT(i,t)$ minimizes the number of unserved jobs in the first i slots, assuming that at most t slots can have load $> H$ and all the other slots have load at most H.

General Uniform Delay D (contd.)

- Define Augmented vector $\mathbf{U^A(i)} = (A(i), \mathbf{U(i)})$.
- Service vector: $\mathbf{S(i)} = (S_0(i), \ldots, S_D(i))$.

Feasible $H \Longrightarrow$

- For an "accounted" slot: $\sum_{j=0}^{D} S_j(i) \le H$
- For a "free" slot: $\sum_{j=0}^{D} S_j(i) \le C$
- Moreover, $0 \le S_j(i) \le U_j^A(i)$

Free Slot i

- Capacity available: C
- Greedily distribute capacity among jobs in $\mathbf{U}^{\mathbf{A}}(i,t) = (A(i), \mathbf{U}(i-1, t-1))$ favoring earlier jobs.
- The jobs unserved form vector $\mathbf{U}^{C}(i,t)$

$$OPT_1(i,t) = \sum_{j=1}^{D} U_j^C(i,t)$$

Accounted Slot i

- Capacity Available: H
- Greedily distribute capacity among jobs in $\mathbf{U}^A(i,t) = (A(i), \mathbf{U}(i-1,t))$ favoring earlier jobs.
- If $U_D^A(i,t) > H$, $OPT_2(i,t) = \infty$
- The jobs unserved form vector $\mathbf{U}^H(i,t)$.

$$OPT_2(i,t) = \sum_{j=1}^{D} U_j^H(i,t)$$

General Uniform Delay D (contd.)

- $OPT(i,t) = \min(OPT_1(i,t), OPT_2(i,t))$
- Correspondingly, $U^C(i,t)$ or $U^H(i,t)$ is retained as $U(i,t)$.
- Therefore, by definition, $OPT(i,t) = \sum_{j=1}^{D} U_j(i,t)$

- Proof is similar to D=1, except now we need to prove that $U_D(i-1,t) \leq U_D^O(i-1,t)\ \forall\, t \leq (i-1)$ for any algorithm O.

Claim: For any (i',t'), if $U_j(i',t') > 0$ for any $1 < j \leq D$, then $U_k(i',t') = A(i'-k+1)\ \forall\, k < j$.

Now we use induction hypothesis to complete the proof.

Online Percentile Problem: Unbounded Competitive Ratio

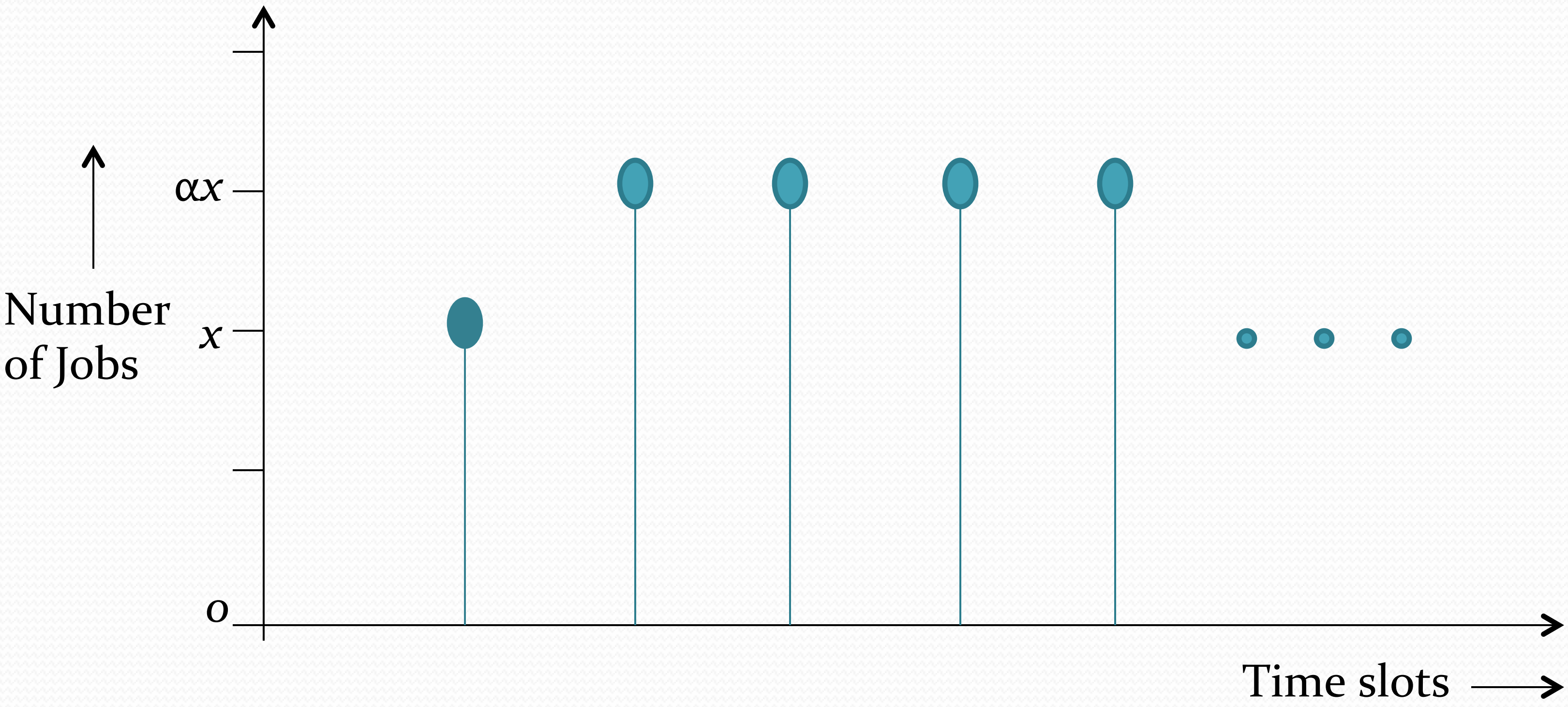
Online Problem: Hardness

- For this version, we assume that $A(i)$ cannot be observed until time slot i.

- We use N_A^X to denote the 95th percentile of the service schedule generated by algorithm X for arrival sequence $A(1)$ through $A(N)$.

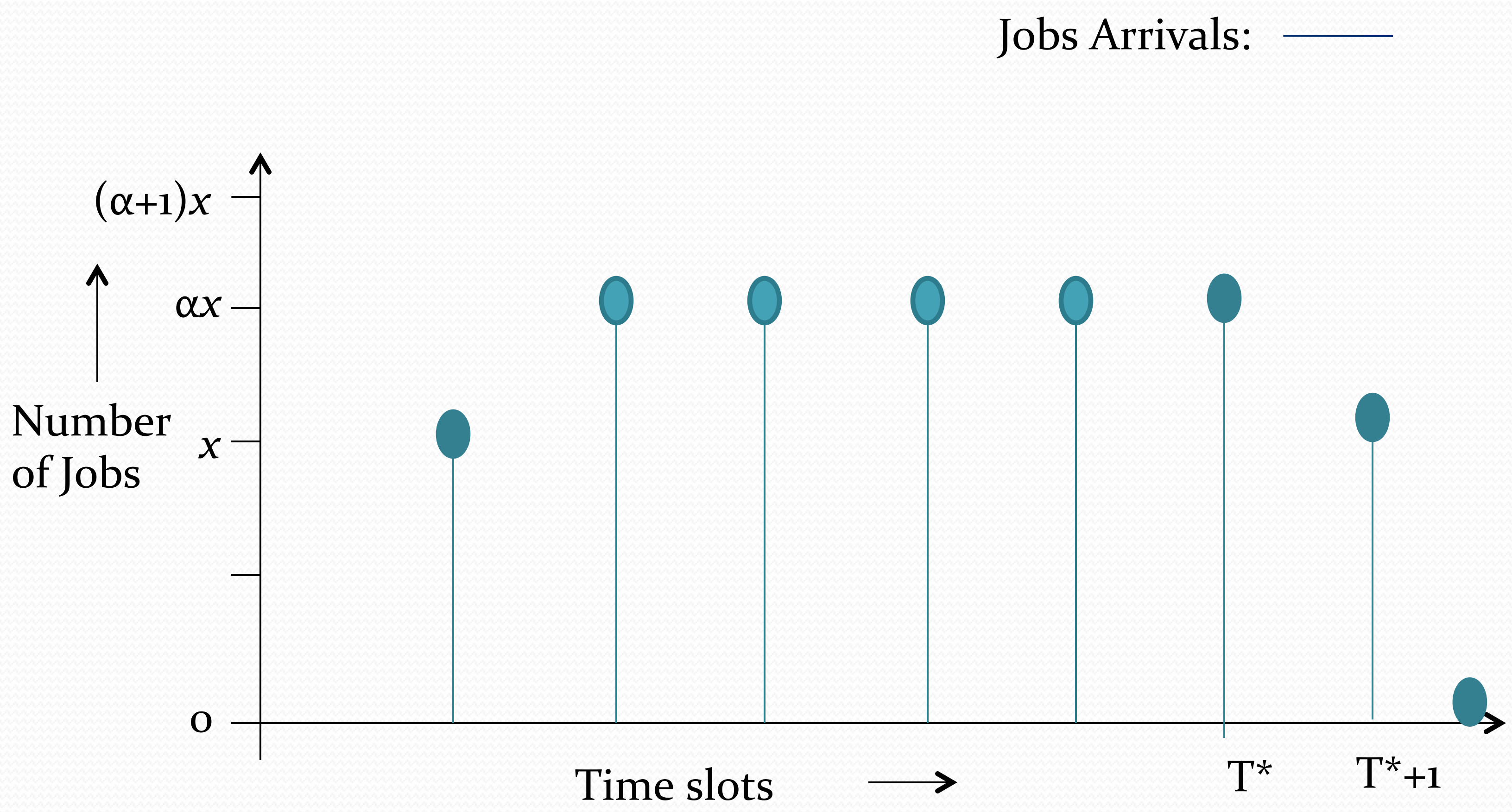
Theorem 3. For any online algorithm OL, there exists an arrival series $A(i), i = 1, 2, ..., N$ for unit delay, such that $\frac{N_A^{OL}}{N_A^{OPT}}$ cannot be bounded by any constant.

Online Problem: Lower Bound Sequence

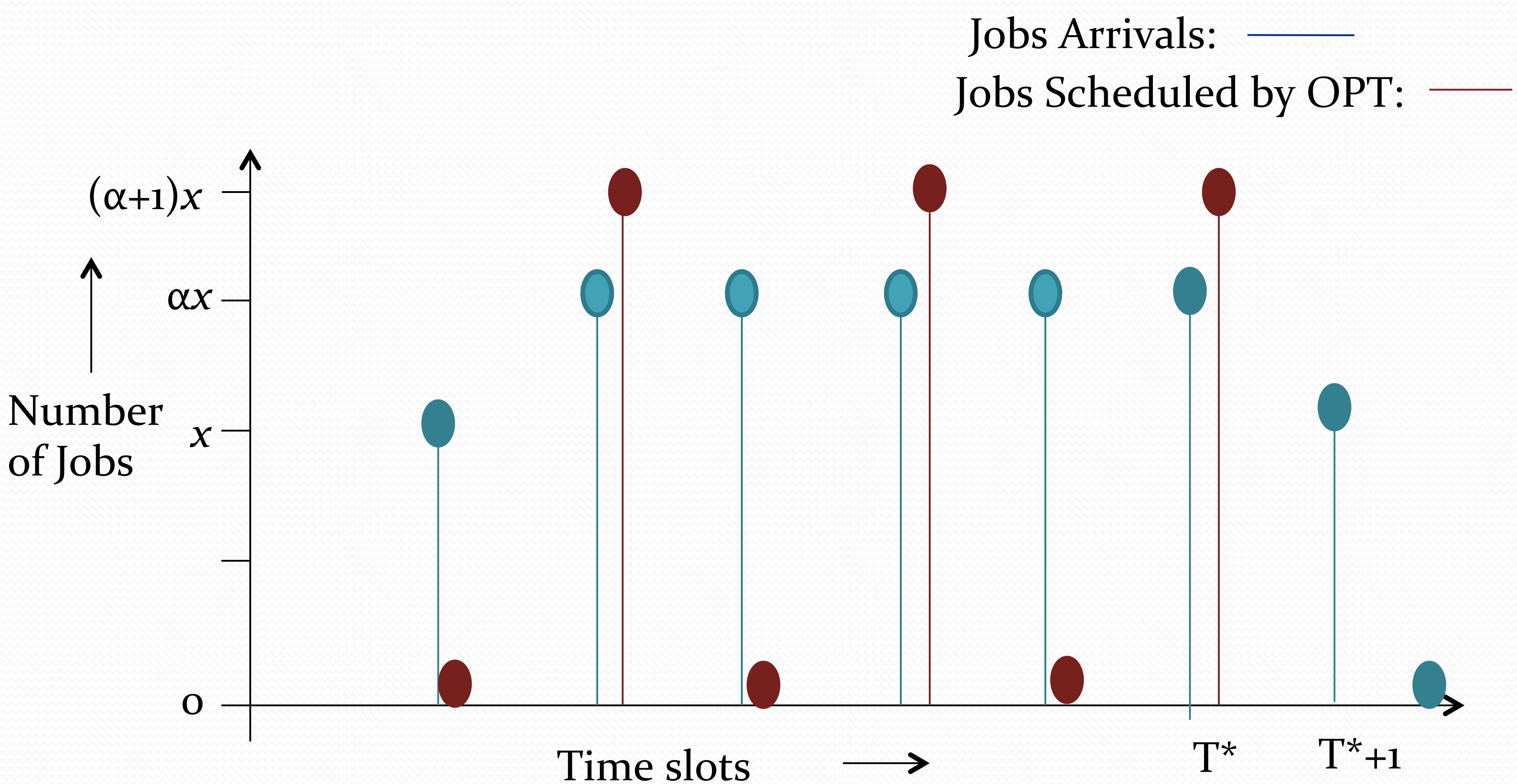


Generate such arrivals until the number of time slots with non-zero job service (S(▪)) is exactly T for the online algorithm. Suppose this happens at time slot $T^* \leq 2T$

Lower Bound Sequence $T^* < 2T$

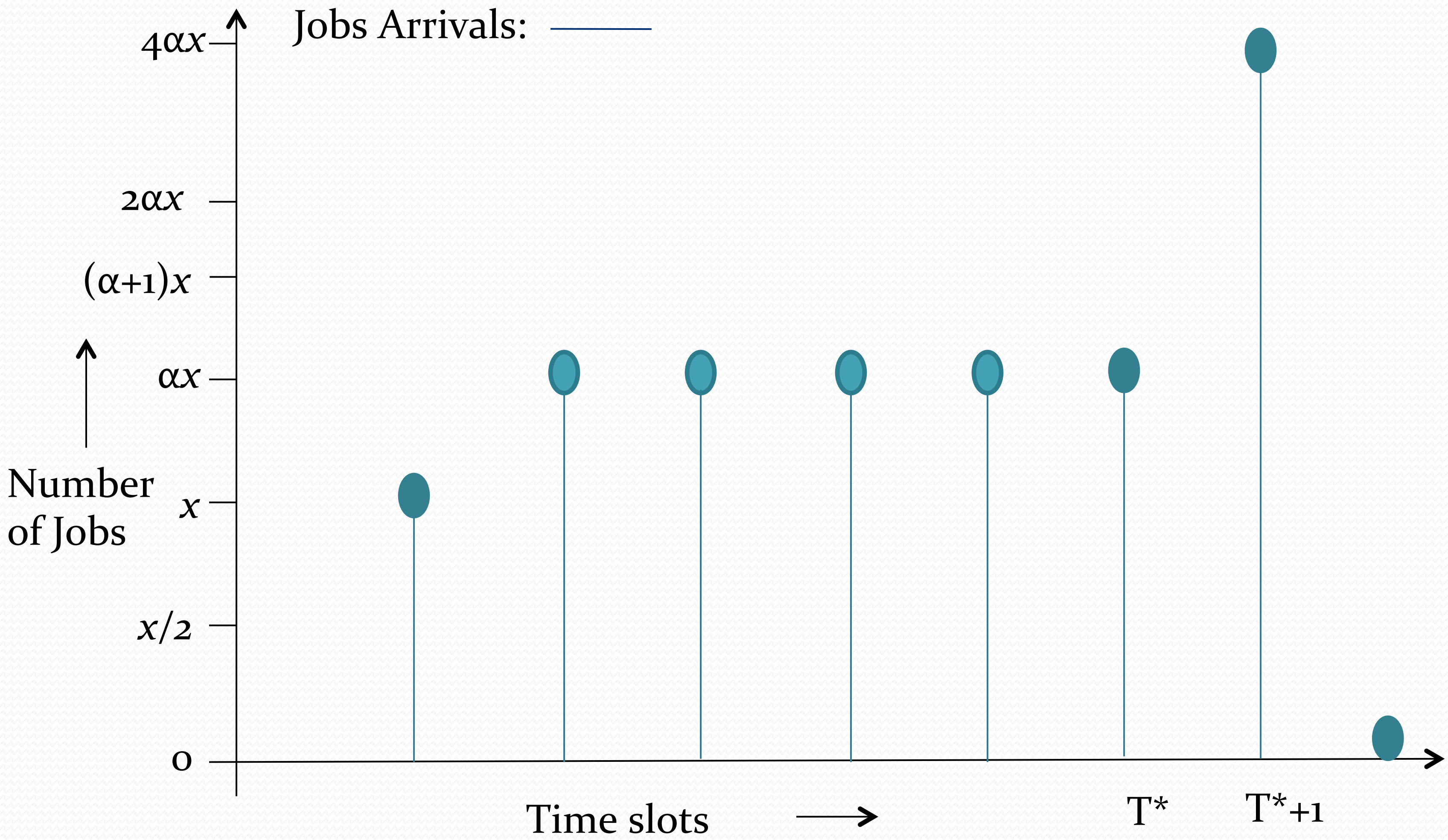


Lower Bound Sequence $T^* < 2T$

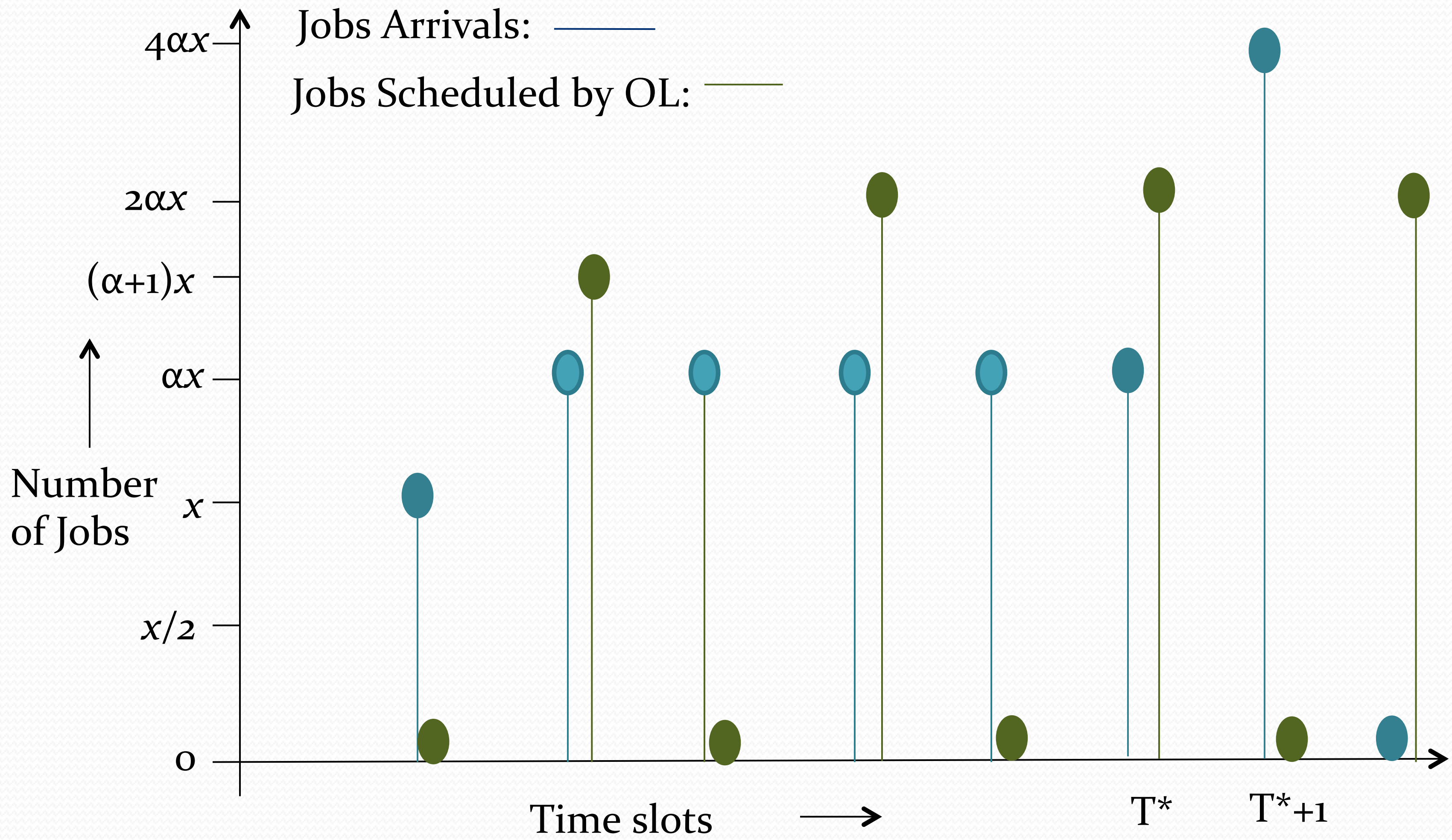


Since $T^* \leq 2T + 1,\ N_A^{OPT} = 0,\ \text{but}\ N_A^{OL} > 0.$

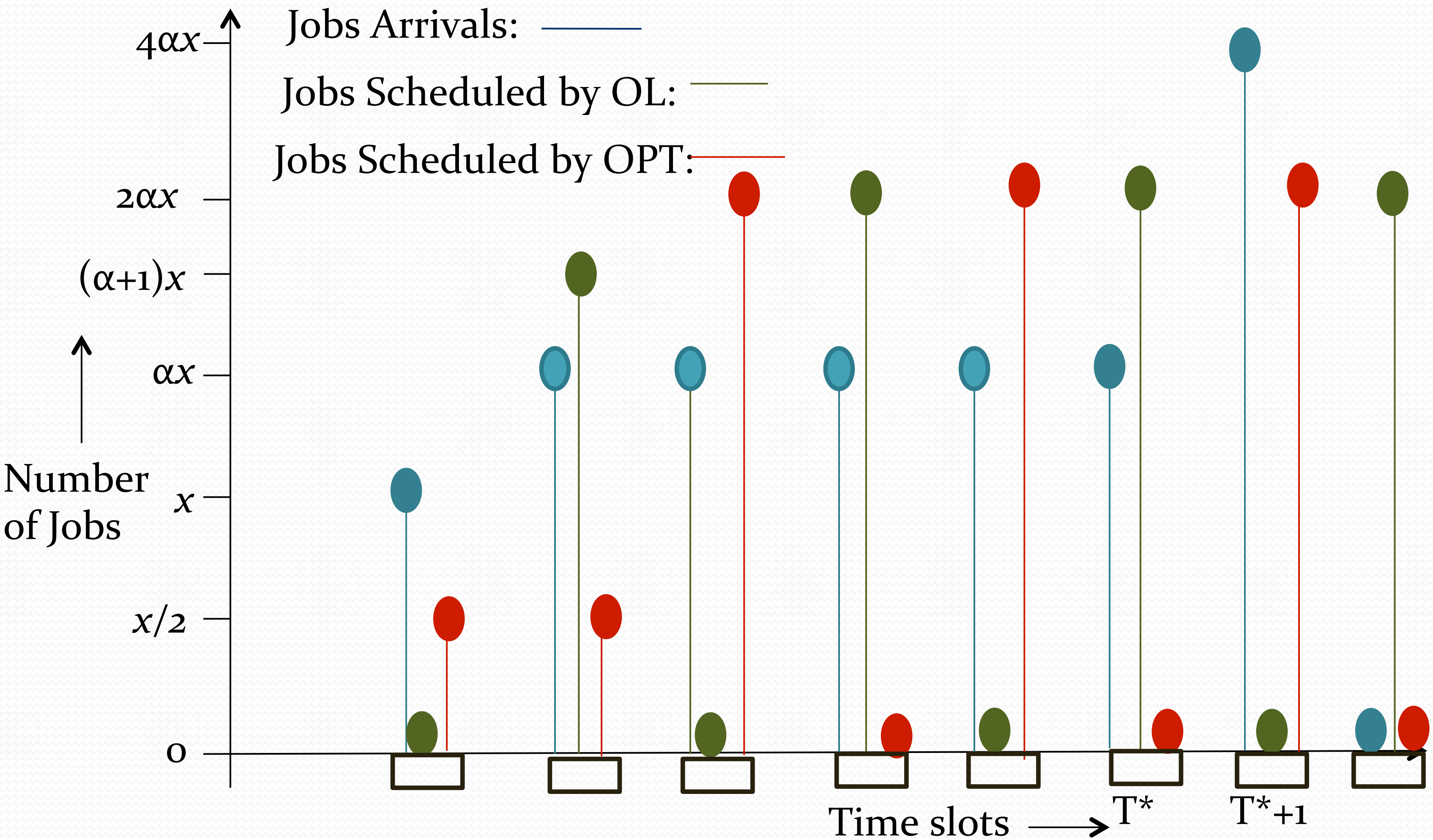
Lower Bound Sequence $T^* = 2T$



Lower Bound Sequence $T^* = 2T$



Lower Bound Sequence $T^* = 2T$



$$N_A^{OPT} = \frac{x}{2}, \text{ but } N_A^{OL} = 2\alpha x$$

Online Min-Max Problem:
Upper Bound and Lower Bound

The Min-Max Problem

- The online percentile problem is inapproximable….
- We consider a related problem: The Min-Max Problem.
- Here we minimize the maximum number of jobs served in a slot, instead of the 95th percentile.

Problem Definition:

- $A(i)$ unit jobs arrive in slot i,
  $\mathbf{S}(i)$: the vector of jobs served in time slot i.
- Jobs can be delayed for at most D time slots.

Goal: Minimize maximum $\sum_j S_j(i)$ over all i.

Related Work

- The offline min-max problem is a special case of the energy minimization problem solved by Yao, Demers and Shenker (FOCS 1995).
- An online algorithm for the general case was provided by Bansal, Kimbrel and Pruhs (FOCS 2004).
- We provide a better approximation for the specific problem we consider.
- We also provide an almost matching lower bound for this specific problem.

Equal Split: An Online Algorithm for Min-Max

Define $A(i) = 0$ when $i \leq 0$.

Algorithm 1 ES: Calculate $S^{ES}(i), i = 1, 2, ...$

1: **for** time slot $i = 1, 2, ...$ **do**
2: Obtain $A(i)$
3: $S(i) \leftarrow \frac{1}{D+1}(A(i-D), A(i-D+1), ..., A(i))$
4: **end for**

Theorem 4. The competitive ratio of ES is $\dfrac{2D+1}{D+1}$.

Lower Bound for Online Min-Max Problem

Theorem 5. For the min-max problem with maximum allowed delay D, the competitive ratio of any deterministic online algorithm denoted by OL is at least $\frac{2D+1}{D+F(D)}$, where, $F(D)=\sum_{i=1}^{D+1}\frac{i}{D+i}$.

[The lower bound construction is non-trivial, and hence not depicted here. Please refer to the paper for details.]

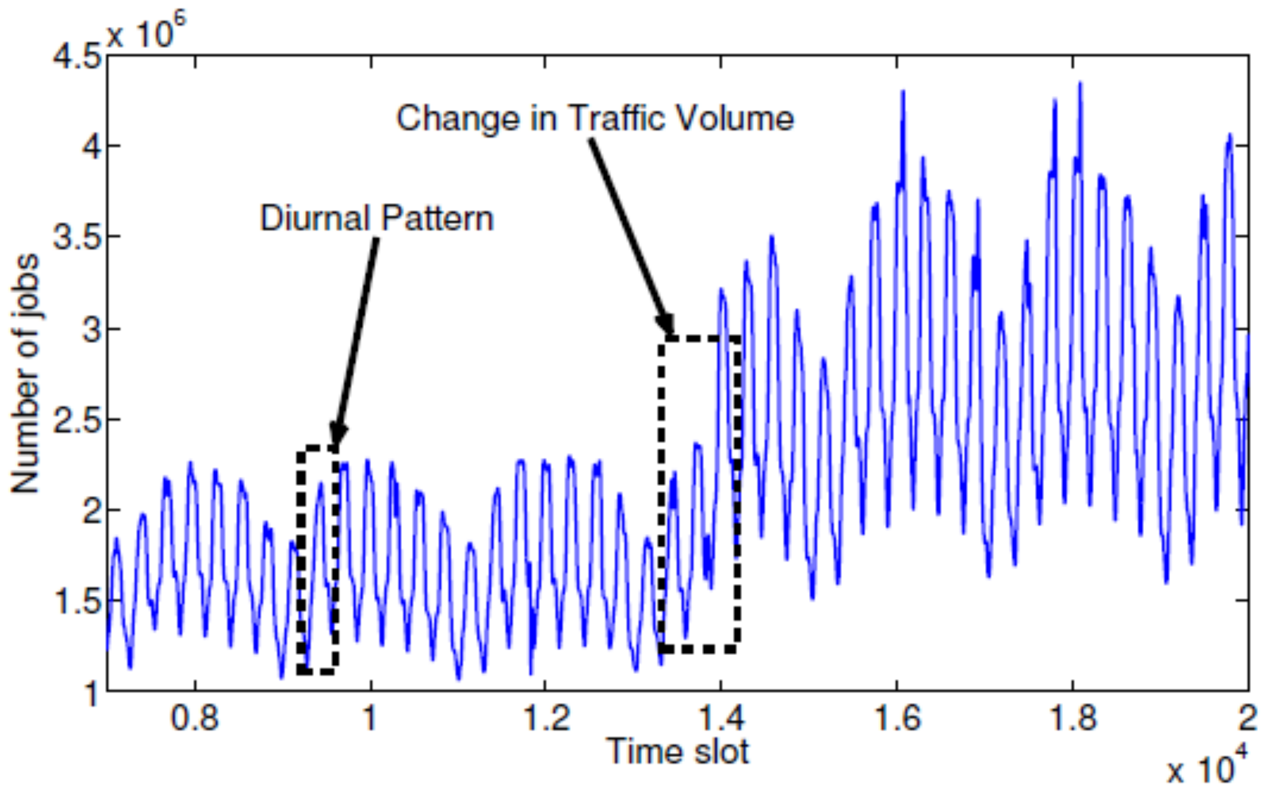
Heuristic Approach

A Heuristic Approach: HELPER

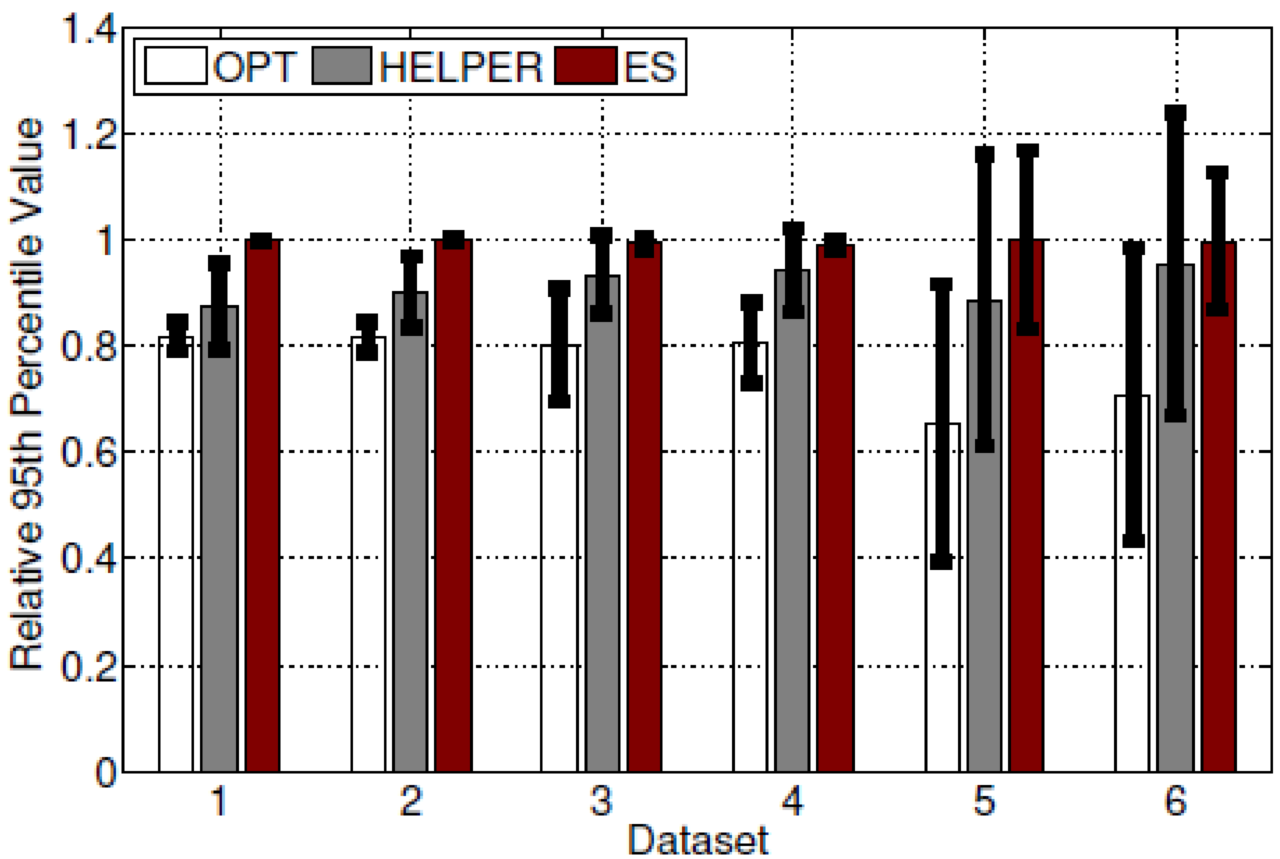
- Traffic in real systems sometime exhibits a repetitive pattern.
- Such correlation can be leveraged to obtain good prediction for the next accounting cycle.
- Based on the above idea, we propose and extensively test a heuristic: $\boxed{\text{HELPER}}$.
- Observe job arrivals for day x, apply OPT, and calculate $\mathbf{S}_x$ and $\mathbf{U}_x$.
- Calculate $r_x(i) = \dfrac{S_x(i) - U_x(i)}{A_x(i)}$.
- On day $(x+1)$, serve either $\boxed{r_x(i) A_{x+1}(i) + U_{x+1}(i-1)}$ or $\boxed{A_{x+1}(i) + U_{x+1}(i-1)}$.

Data Set Used

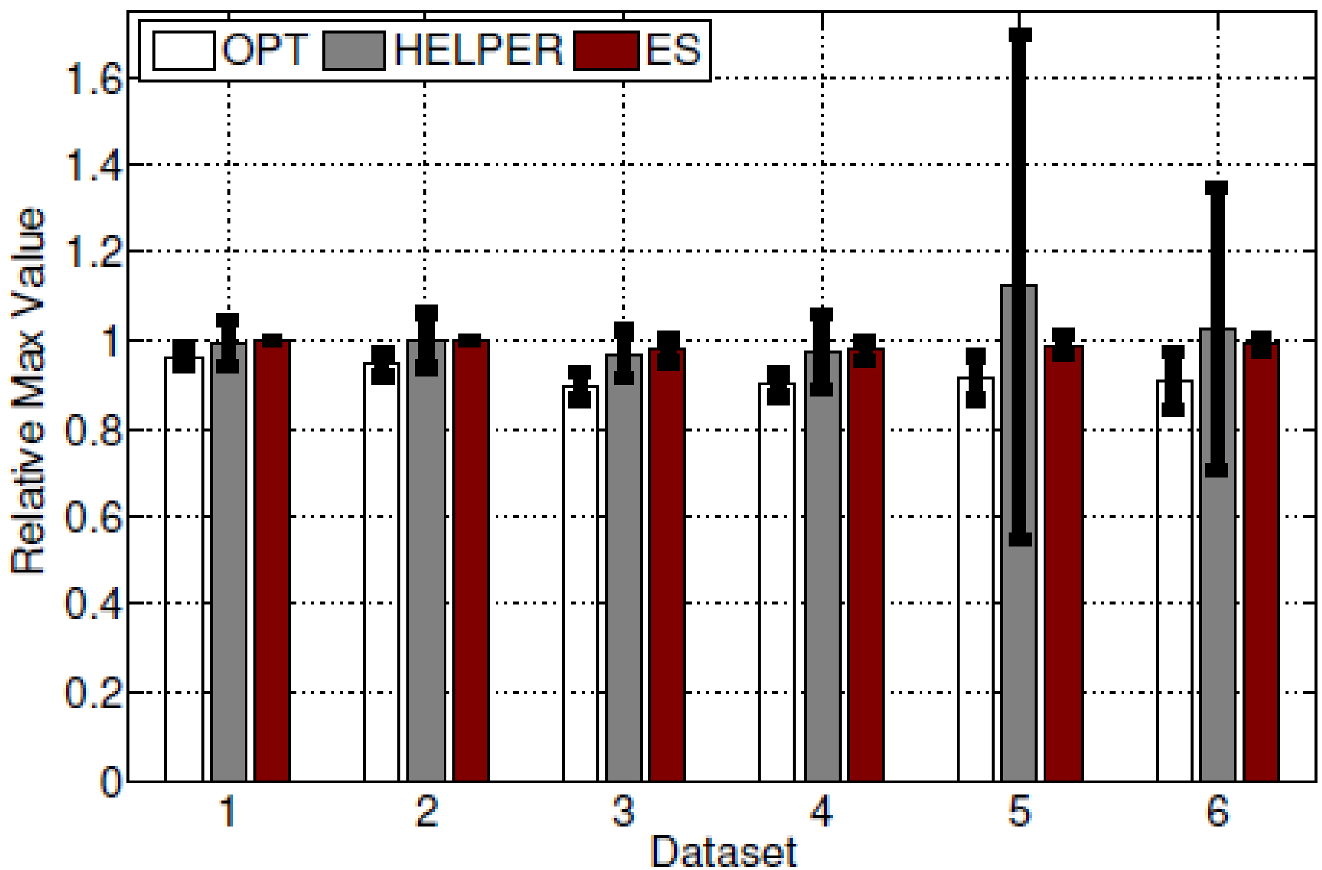
We generated 6 times series with different characteristics using the Wikipedia hit records as the basis. An example shown below:



Experiments: Comparing OPT, ES and HELPER for Percentile Problem



Experiments: Comparing OPT, ES and HELPER for Min-Max Problem



Concluding Remarks

Conclusion and Future Work

We have shown…

- Optimal offline solutions for the percentile problem.
- Online algorithm for the min-max problem.
- Lower bounds for online versions of both problems.

Several interesting questions remain…

- Resource augmentation
- Varying delays
- Bicriteria optimization
- Randomized algorithms

Thank You!