max planck institut
informatik

Online Checkpointing

with

Improved Worst-Case Guarantees

Karl Bringmann Benjamin Doerr Adrian Neumann
Jakub Sliacan

Max Planck Institute for Informatics

June 24, 2013

Problem

We have a *long running computation* and want to *look at the past*.

Problem

We have a *long running computation* and want to *look at the past*.

- Recompute from the beginning
- Store everything

Problem

We have a *long running computation* and want to *look at the past*.

- Recompute from the beginning
- Store everything
- Store checkpoints

Model

- We have storage for k checkpoints
- Only one operation: Delete an old checkpoint and save the current state

Model

- We have storage for k checkpoints
- Only one operation: Delete an old checkpoint and save the current state
- Try to keep the longest interval short

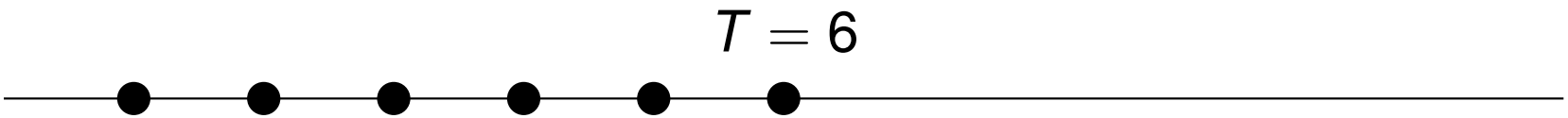
Model

- We have storage for k checkpoints
- Only one operation: Delete an old checkpoint and save the current state
- Try to keep the longest interval short

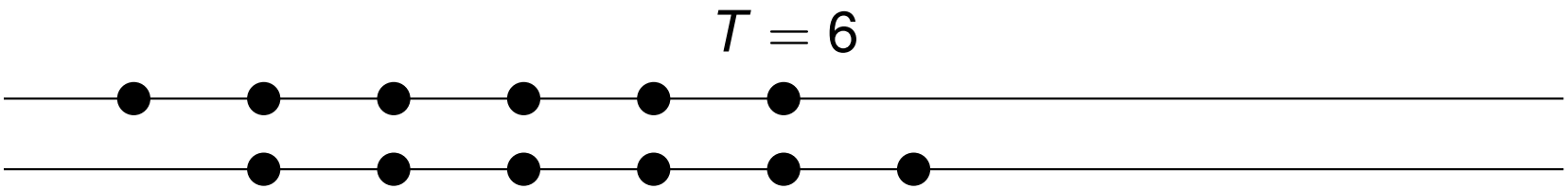
$$\text{Discr}(A) := (k+1) \sup_{T \geq t_k} \frac{\bar{\ell}_T}{T}$$

$\bar{\ell}_T$ is the longest interval at time T.
$T/(k+1)$ is the size of an interval in the equidistant setting.

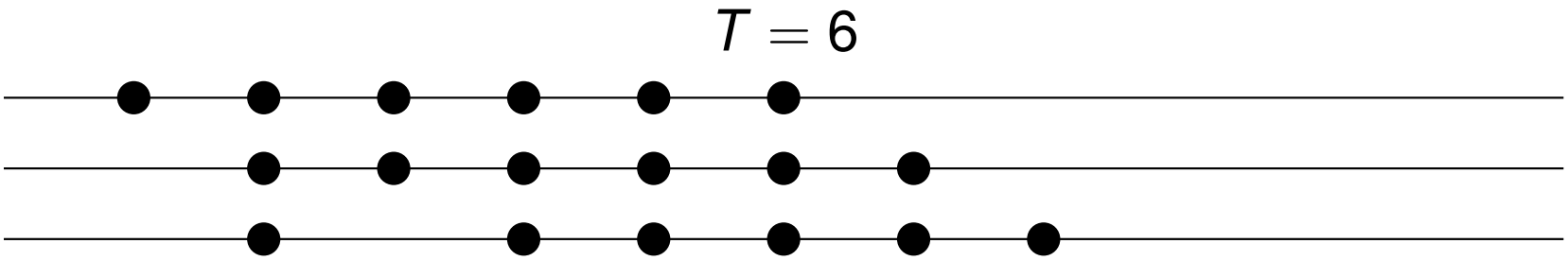
Example $k = 6$



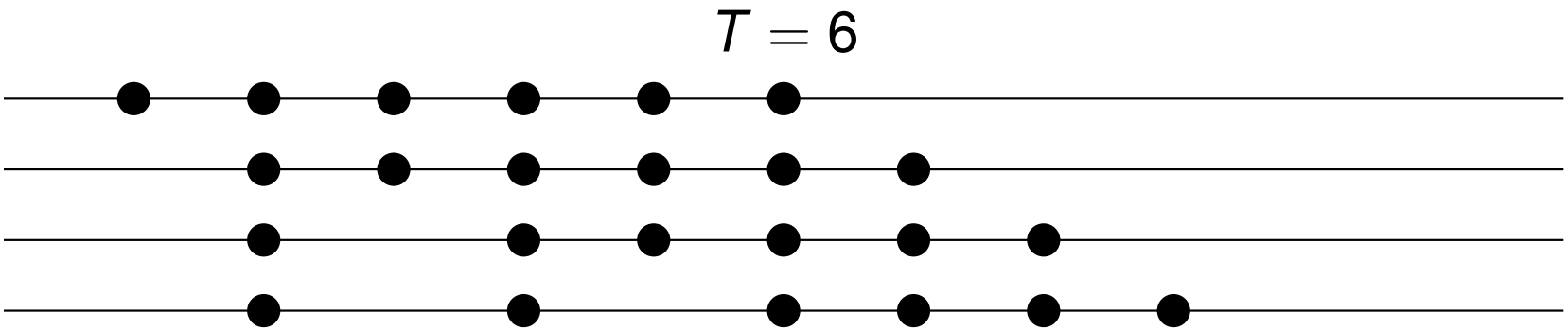
Example $k = 6$



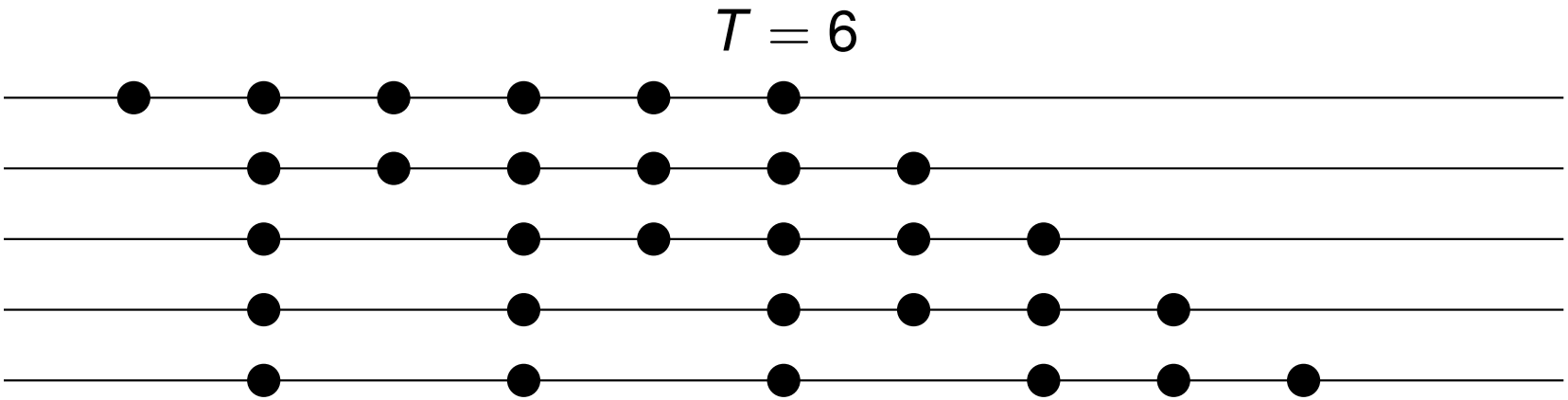
Example $k = 6$



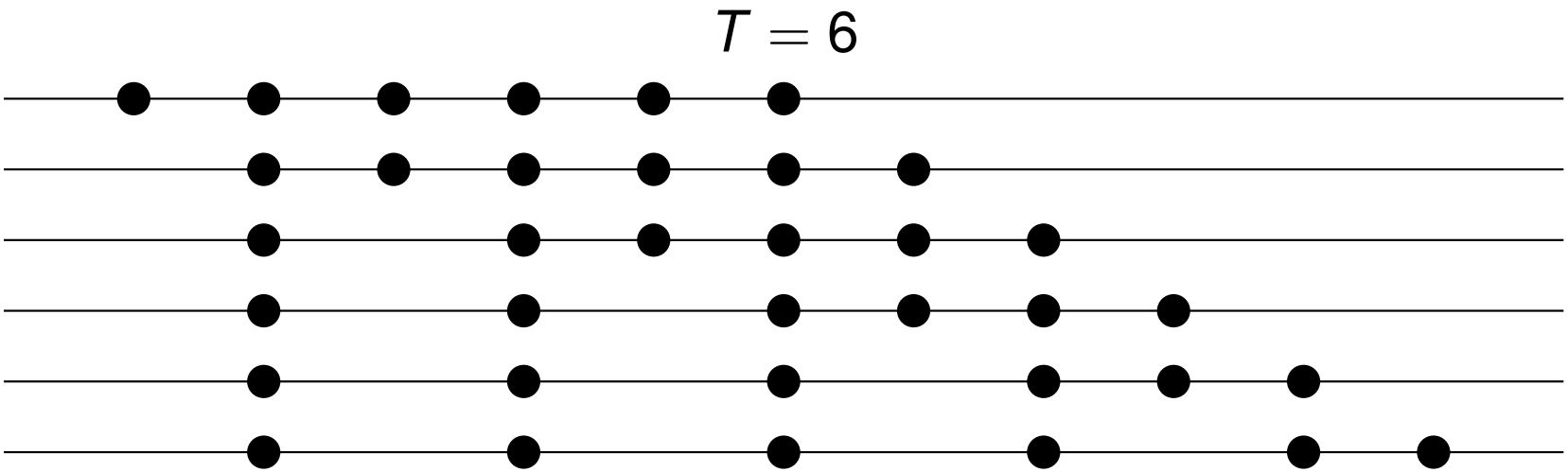
Example $k = 6$



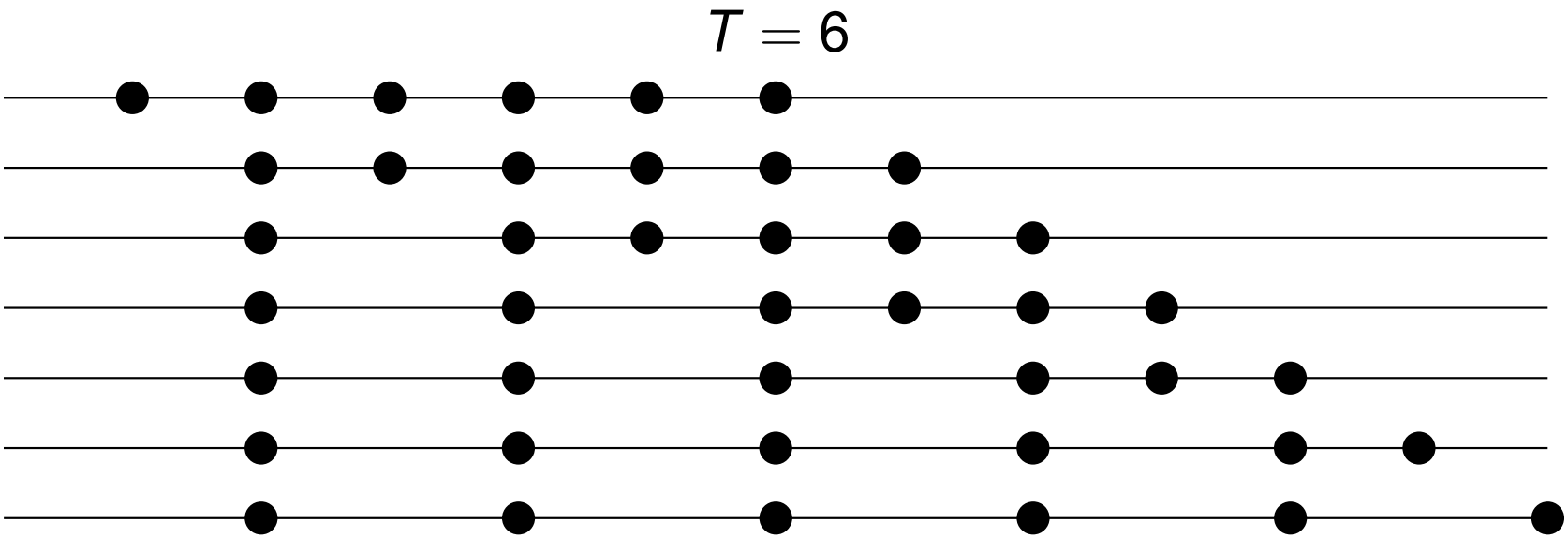
Example $k = 6$



Example $k = 6$



Example $k = 6$



Results

Ahlroth, Pottonen, Schumacher (COCOON '11) show $1 + 1/k \leq p_k \leq 2$. We improve to

$$1.3 \leq p_k \leq 1.6$$

Results

Ahlroth, Pottonen, Schumacher (COCOON '11) show $1 + 1/k \leq p_k \leq 2$. We improve to

$$1.3 \leq p_k \leq 1.6$$

- Algorithms
  - LINEAR: For all k,

$$\text{Discr}(\text{LINEAR}) = 1.59 + O(k^{-1})$$

  - BINARY: For powers of two,

$$\text{Discr}(\text{BINARY}) = \ln(4) + o(1) \approx 1.39$$

  - Results for fixed k: $p_k < 1.5$ for $k \in [5, 60]$
- Lower Bound: $p_k \geq 2 - \ln(2) - o(1) \approx 1.3$

Analysis Technique

We only consider cyclic algorithms

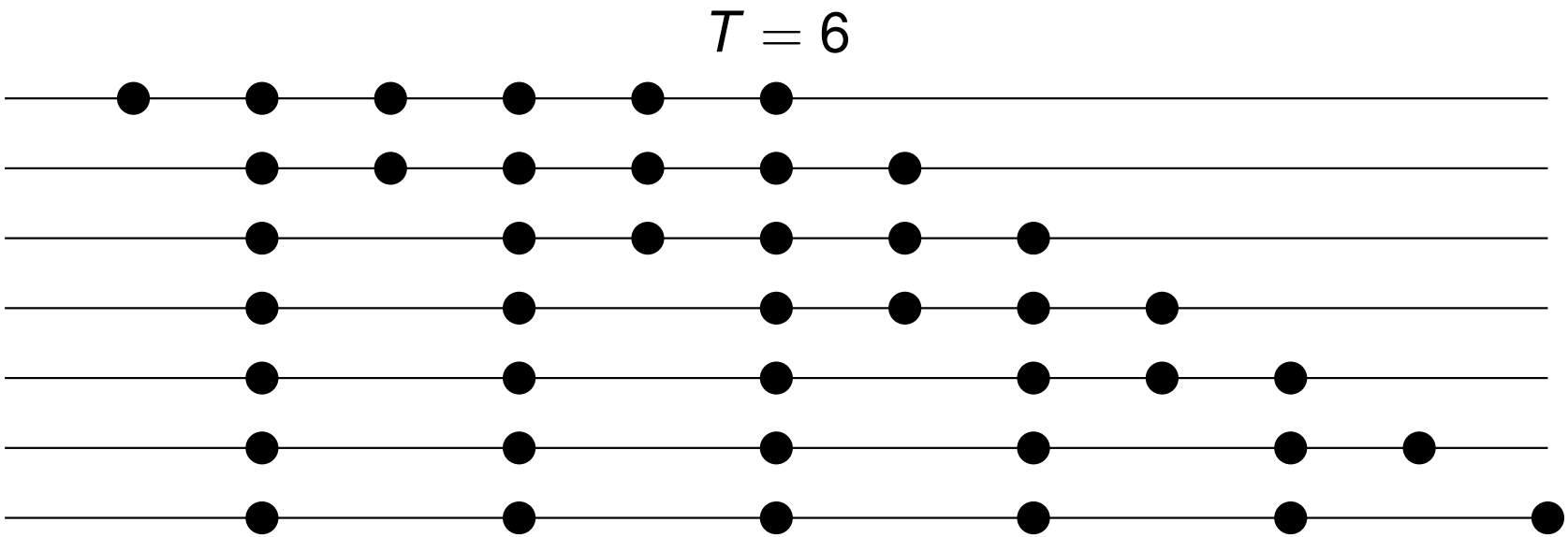
- Remove old checkpoints in a fixed pattern
- After one period all intervals are scaled by some fixed γ

Analysis Technique

We only consider cyclic algorithms

- Remove old checkpoints in a fixed pattern
- After one period all intervals are scaled by some fixed γ

Previous example was cyclic with pattern $P = (1, 2, 3, 4, 5, 6)$ and $\gamma = 2$.



Analysis Technique

We only consider cyclic algorithms

- Remove old checkpoints in a fixed pattern
- After one period all intervals are scaled by some fixed γ

Previous example was cyclic with pattern $P = (1, 2, 3, 4, 5, 6)$ and $\gamma = 2$.

Performance

For cyclic algorithms it suffices to look at one period to find the performance.

Example $k = 3$

Pattern $P = (1)$, $t_3 = 1$

- After one period: Intervals scaled by γ

Example $k = 3$

Pattern $P = (1)$, $t_3 = 1$

- After one period: Intervals scaled by γ

$$t_4 = \gamma \quad t_3 = 1 \quad t_2 = \frac{1}{\gamma} \quad t_1 = \frac{1}{\gamma^2}$$

Example $k = 3$

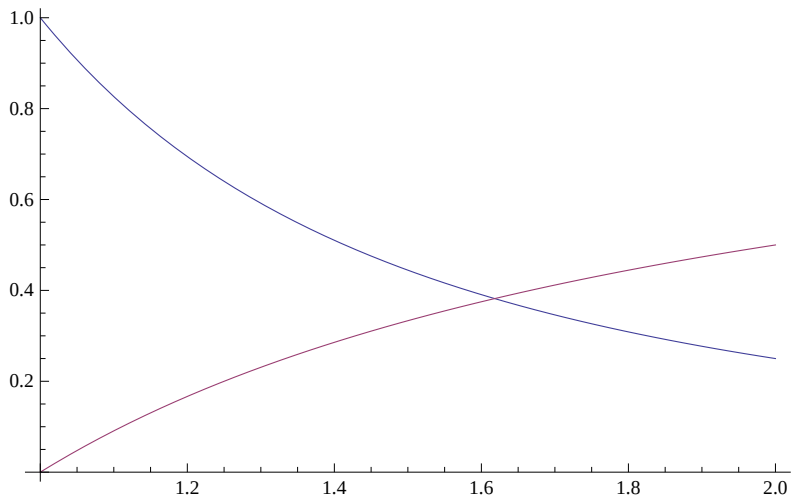
Pattern $P = (1)$, $t_3 = 1$

- After one period: Intervals scaled by γ

$$t_4 = \gamma \quad t_3 = 1 \quad t_2 = \frac{1}{\gamma} \quad t_1 = \frac{1}{\gamma^2}$$

- Intervals to consider

$$\left\{ \frac{t_1 - 0}{t_3}, \frac{t_2 - t_1}{t_3}, \frac{t_3 - t_2}{t_3} \right\}$$
$$\cup \left\{ \frac{t_2 - 0}{t_4}, \frac{t_3 - t_2}{t_4}, \frac{t_4 - t_3}{t_4} \right\}$$
$$= \left\{ \frac{1}{\gamma^2}, \frac{\gamma - 1}{\gamma^2}, \frac{\gamma - 1}{\gamma} \right\}$$



Example $k = 3$

Pattern $P = (1)$, $t_3 = 1$

- After one period: Intervals scaled by γ

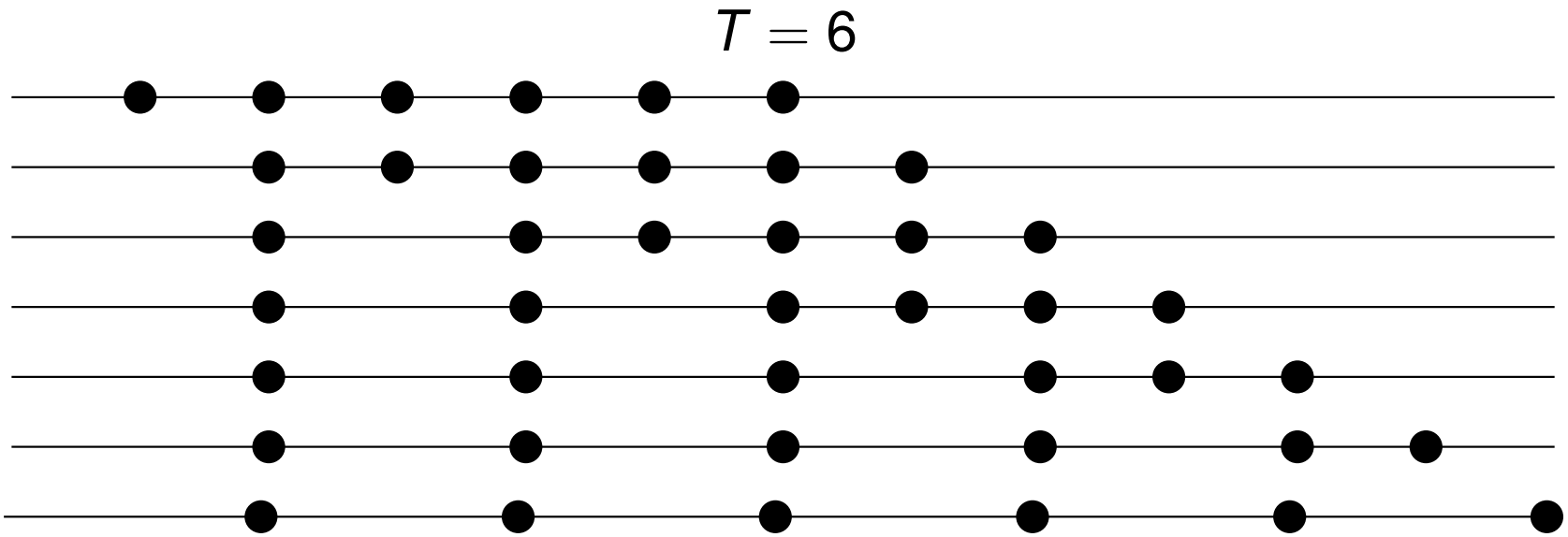
$$t_4 = \gamma \quad t_3 = 1 \quad t_2 = \frac{1}{\gamma} \quad t_1 = \frac{1}{\gamma^2}$$

- Intervals to consider

$$\left\{ \frac{t_1 - 0}{t_3}, \frac{t_2 - t_1}{t_3}, \frac{t_3 - t_2}{t_3} \right\}$$
$$\cup \left\{ \frac{t_2 - 0}{t_4}, \frac{t_3 - t_2}{t_4}, \frac{t_4 - t_3}{t_4} \right\}$$
$$= \left\{ \frac{1}{\gamma^2}, \frac{\gamma - 1}{\gamma^2}, \frac{\gamma - 1}{\gamma} \right\}$$

Performance: $4 * 1/\phi^2 \approx 1.53$

First Example Again



LINEAR

Pattern $P = (1, 2, \ldots, k)$. Checkpoints placed at times

$$t_i = (i/k)^\alpha \qquad \alpha = 1.302$$

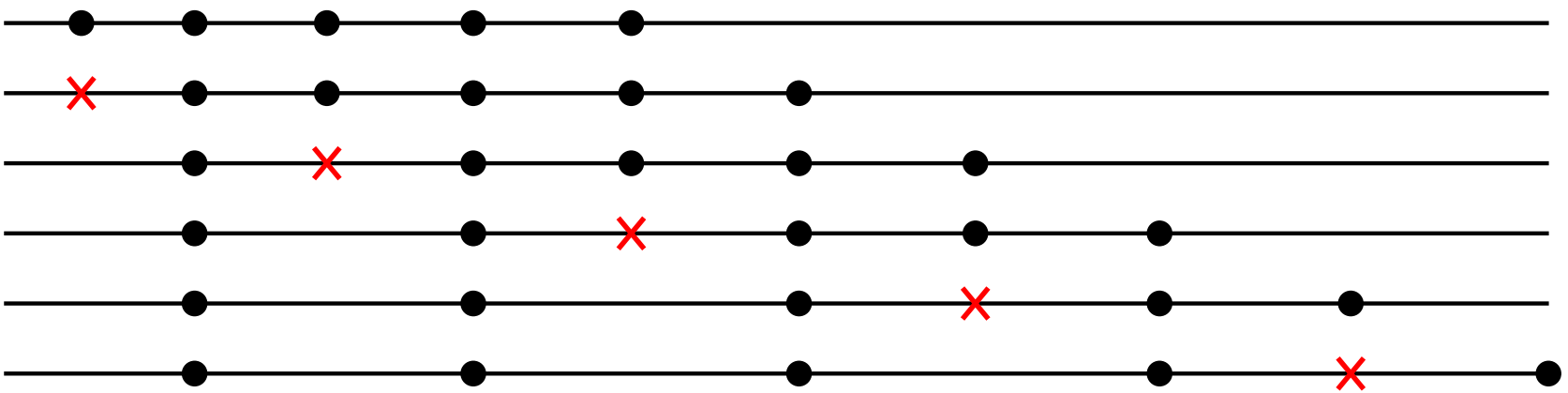
- Works for all k
- Asymptotic performance $1.59 - O(k^{-1})$

LINEAR

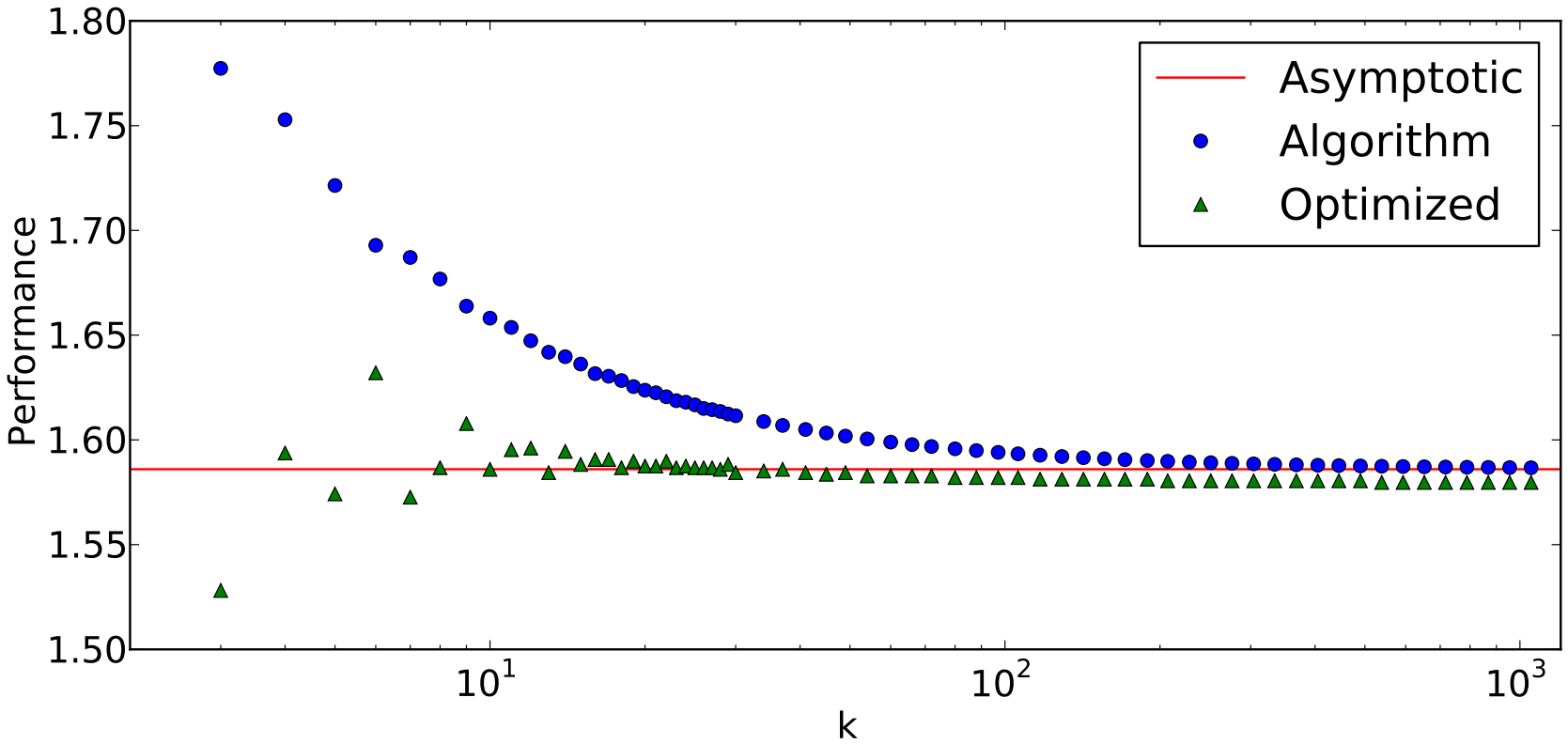
Pattern $P = (1, 2, \ldots, k)$. Checkpoints placed at times

$$t_i = (i/k)^\alpha \qquad \alpha = 1.302$$

- Works for all k
- Asymptotic performance $1.59 - O(k^{-1})$



Experiment


1.80
1.75
1.70
1.65
1.60
1.55
1.50
Performance
10^1
10^2
10^3
k
Asymptotic
Algorithm
Optimized

BINARY

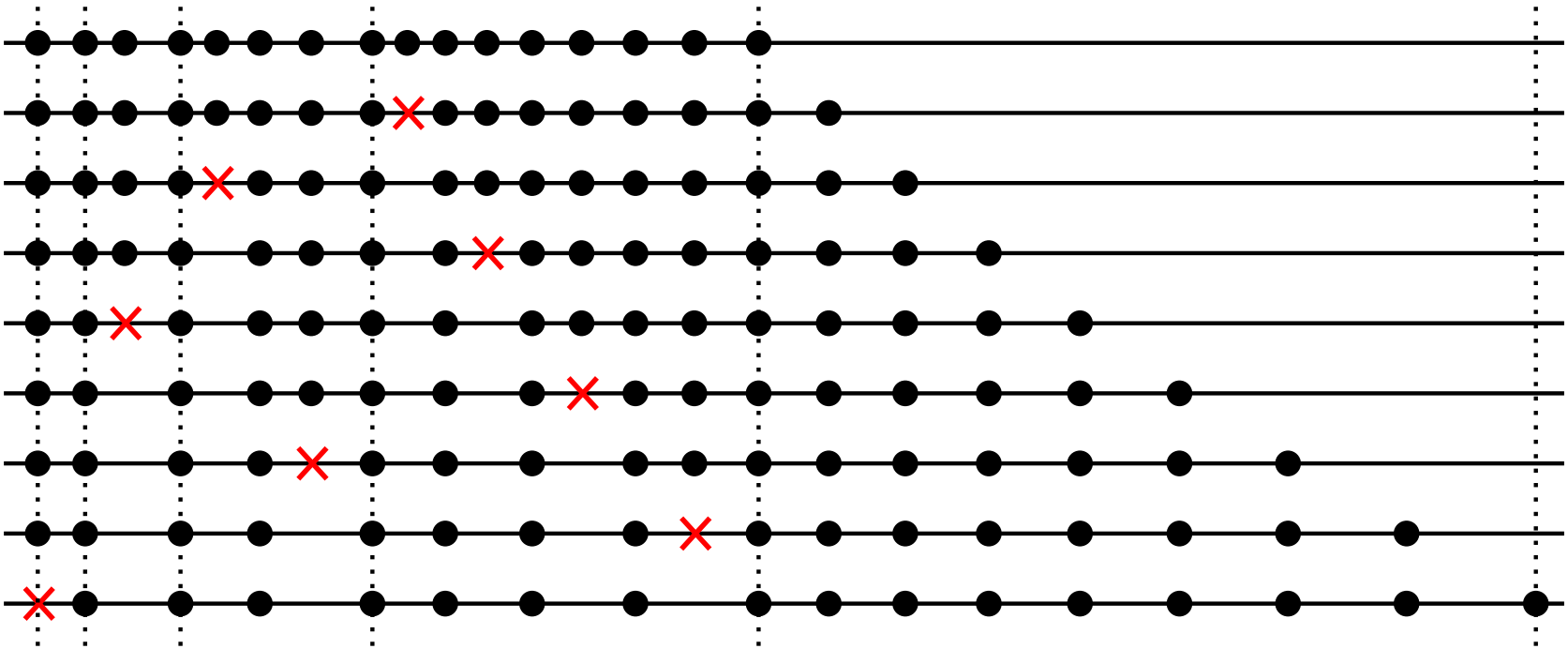
Works for powers of two.

$$\mathrm{Discr}(\textsc{Binary}) = \ln 4 - o(1) \approx 1.39$$

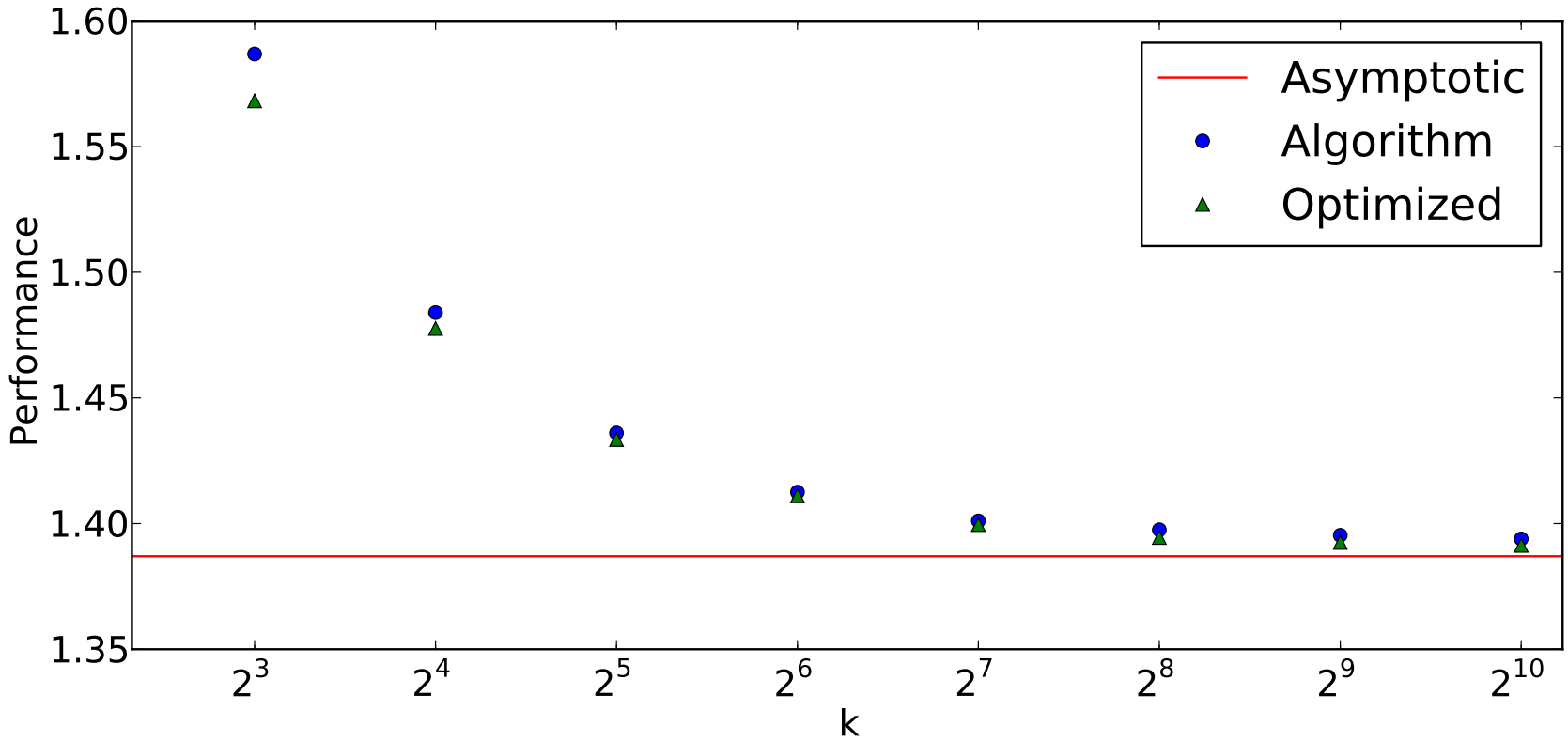
BINARY

Works for powers of two.

$$\mathrm{Discr}(\textsc{Binary}) = \ln 4 - o(1) \approx 1.39$$



Experiment


1.60
1.55
1.50
1.45
1.40
1.35
Performance
2^3
2^4
2^5
2^6
2^7
2^8
2^9
2^{10}
k
Asymptotic
Algorithm
Optimized

Fixed k

For fixed k we find good algorithms via Linear Programming.
Assume P and γ are fixed.

Fixed k

For fixed k we find good algorithms via Linear Programming.
Assume P and γ are fixed.

- Order the t_i

$$t_i \leq t_{i+1},$$

Fixed k

For fixed k we find good algorithms via Linear Programming.
Assume P and γ are fixed.

- Order the t_i

$$t_i \leq t_{i+1},$$

- Enforce scaling factor

$$\tau_i^n = \gamma \tau_i^0.$$

Fixed k

For fixed k we find good algorithms via Linear Programming. Assume P and γ are fixed.

- Order the t_i

$$t_i \leq t_{i+1},$$

- Enforce scaling factor

$$\tau_i^n = \gamma \tau_i^0.$$

- Enforce a performance of λ

$$\tau_{i+1}^j - \tau_i^j \leq \lambda \tau_k^j / (k+1)$$

Fixed k

Find algorithms by

- Binary Search over λ
- Linear Search over γ
- Trying different patterns

Fixed k

Find algorithms by

- Binary Search over λ
- Linear Search over γ
- Trying different patterns

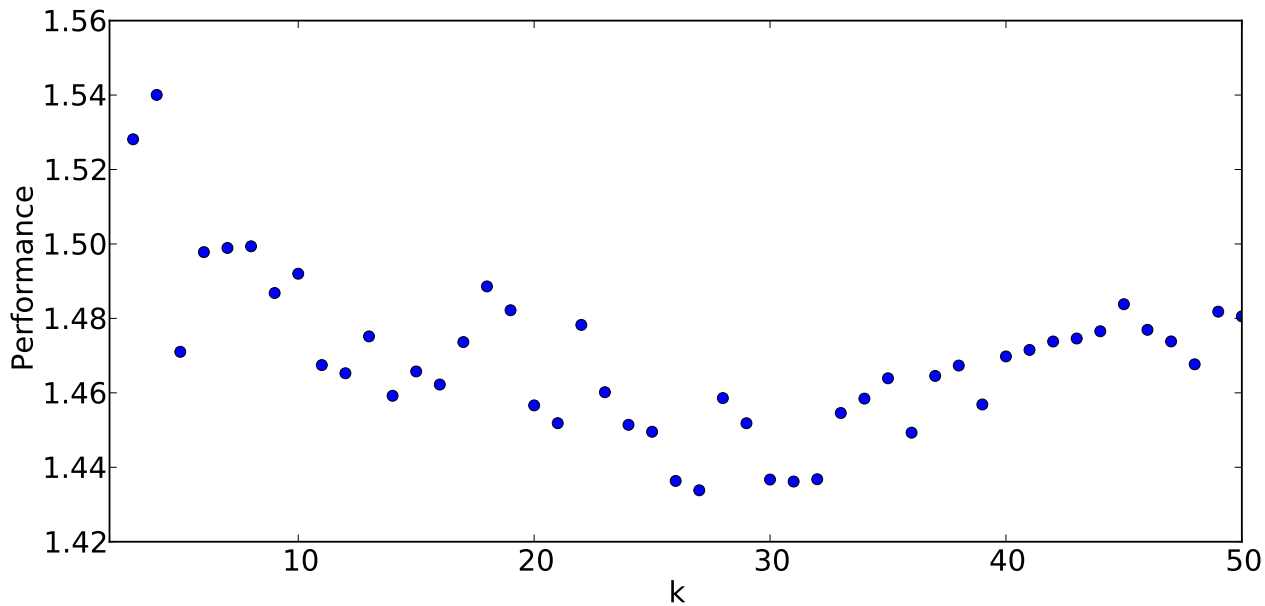
k	3	4	5	6	7	8
Perf.	1.53	1.54	1.47	1.49	1.49	1.49

Fixed k

Find algorithms by

- Binary Search over λ
- Linear Search over γ
- Trying different patterns

k	3	4	5	6	7	8
Perf.	1.53	1.54	1.47	1.49	1.49	1.49



Lower Bound

We show the first lower bound that is asymptotically greater than 1

$$p_k \geq 1.3 - O(k^{-1})$$

Lower Bound

We show the first lower bound that is asymptotically greater than 1

$$p_k \geq 1.3 - O(k^{-1})$$

Proof Sketch

Let A be an algorithm with performance p.

- Analyze A until $k/(2p)$ initial checkpoints are removed
- Assume intervals from storing new points are always optimal
- Bound the sizes of
  - Unchanged intervals
  - Intervals from deletion
  - Intervals from insertion

Optimal Algorithms Exist

$$p_k = \inf_{A_k} \mathrm{Discr}(A_k)$$

Optimal Algorithms Exist

$$p_k = \inf_{A_k} \mathrm{Discr}(A_k)$$

The infimum can be replaced by a minimum.

Proof sketch

- Starting positions with quality p_k exist
- Two algorithms can be combined losslessly
- There always are algorithms that make a few steps without worsening quality
- Combine a sequence of such algorithms to get optimal A^*.

Summary

We show for all k

$$1.3 \leq p_k \leq 1.6$$

- First lower bound asymptotically larger than one.
- First upper bound asymptotically smaller than two.

For $k = 2^i$ we show

$$1.3 \leq p_k \leq 1.39$$

only about 6% room for improvement left!

Open Problems

- Different performance measures, e.g. expected recomputation time
- Analysis of greedy algorithms