

Evaluating Cooperative Checkpointing for Supercomputing Systems



Adam J. Oliner, Stanford University



Ramendra K. Sahoo, IBM, T.J. Watson

April 29th

SMTPS 2006, Rhodes, Greece



The Goal

- Evaluate the applicability of cooperative checkpointing to large-scale systems through realistic simulations

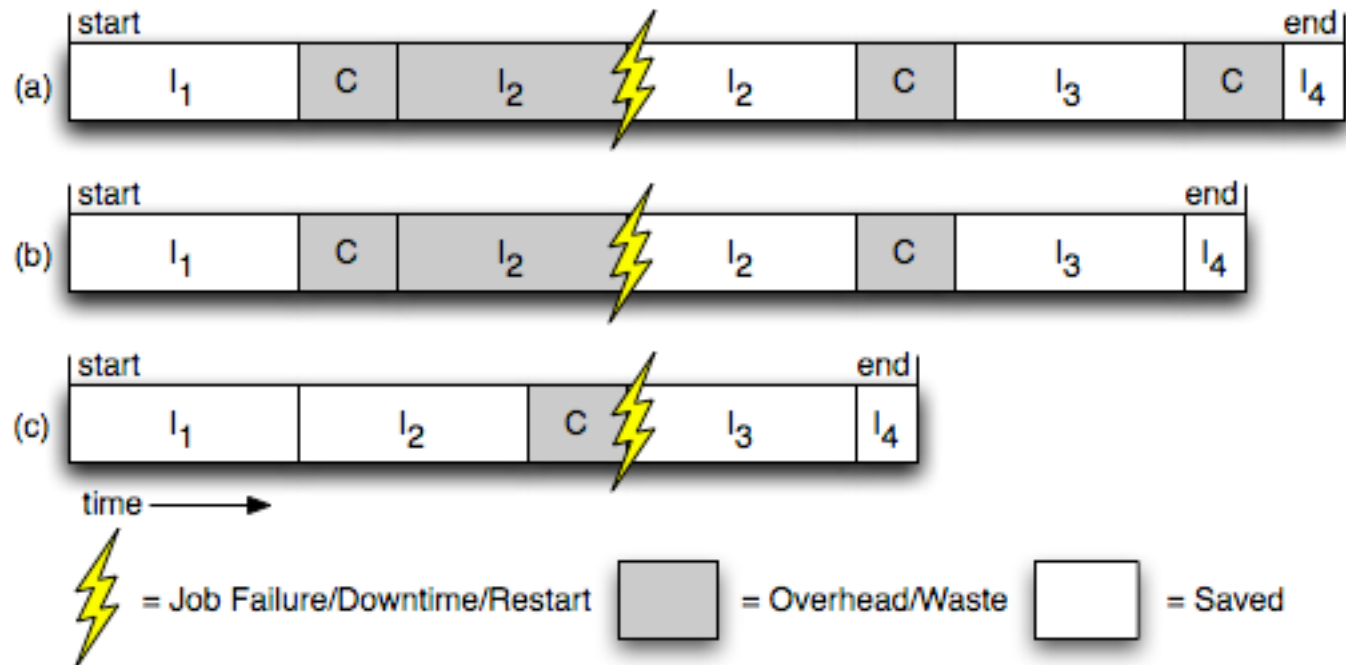


Cooperative Checkpointing

1. User inserts checkpoint requests
2. Compiler optimizes
3. Runtime gatekeeper grants/denies
 - Online heuristics = CC algorithm



CC Example



- Intuition: skip checkpoints less likely to be used for rollback



Heuristics

- Perform or skip?
- Work-based
 - Unsaved work \geq checkpoint overhead
 - $dl \geq C_i$
- Risk-based
 - Expected loss \geq checkpoint overhead
 - $p_f dl \geq C_i$



Simulations

- 128 nodes
- Topology: {3D torus, flat}
- Job logs: {NASA, SDSC, LLNL}
- Failures: AIX cluster over 1 year
- C: {720, 3600} seconds
- I: {500, 1000, ..., 30500} seconds



Metrics

- Bounded slowdown
 - Turnaround time relative to optimal

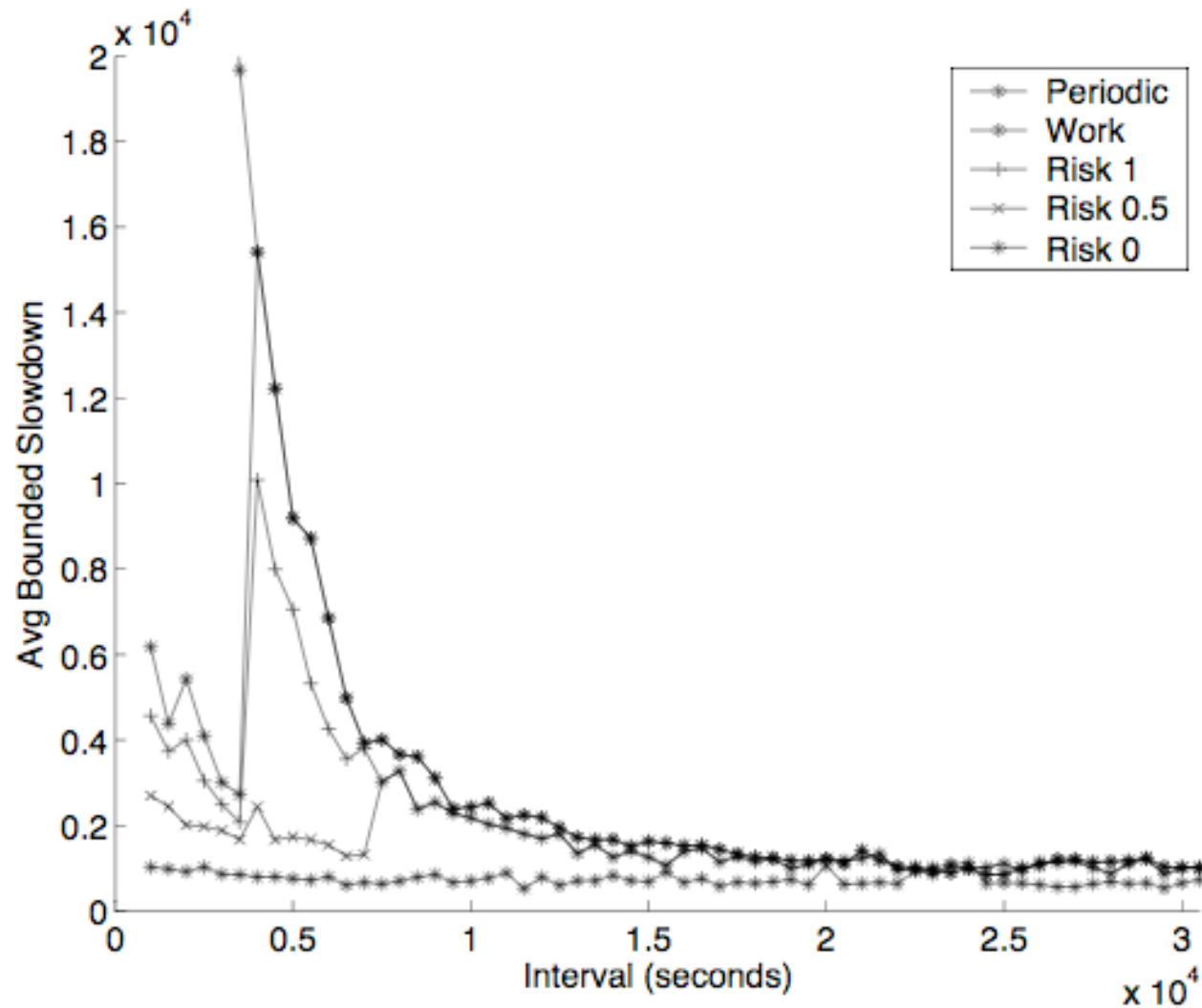
$$\frac{\max(r_j, \Gamma)}{\max(e_j, \Gamma)}, \quad \Gamma = 10 \text{ seconds}$$

- Lost work
 - Time spent recomputing

$$\sum_{\forall x} (t_x - c_{jx}) n_{jx}$$

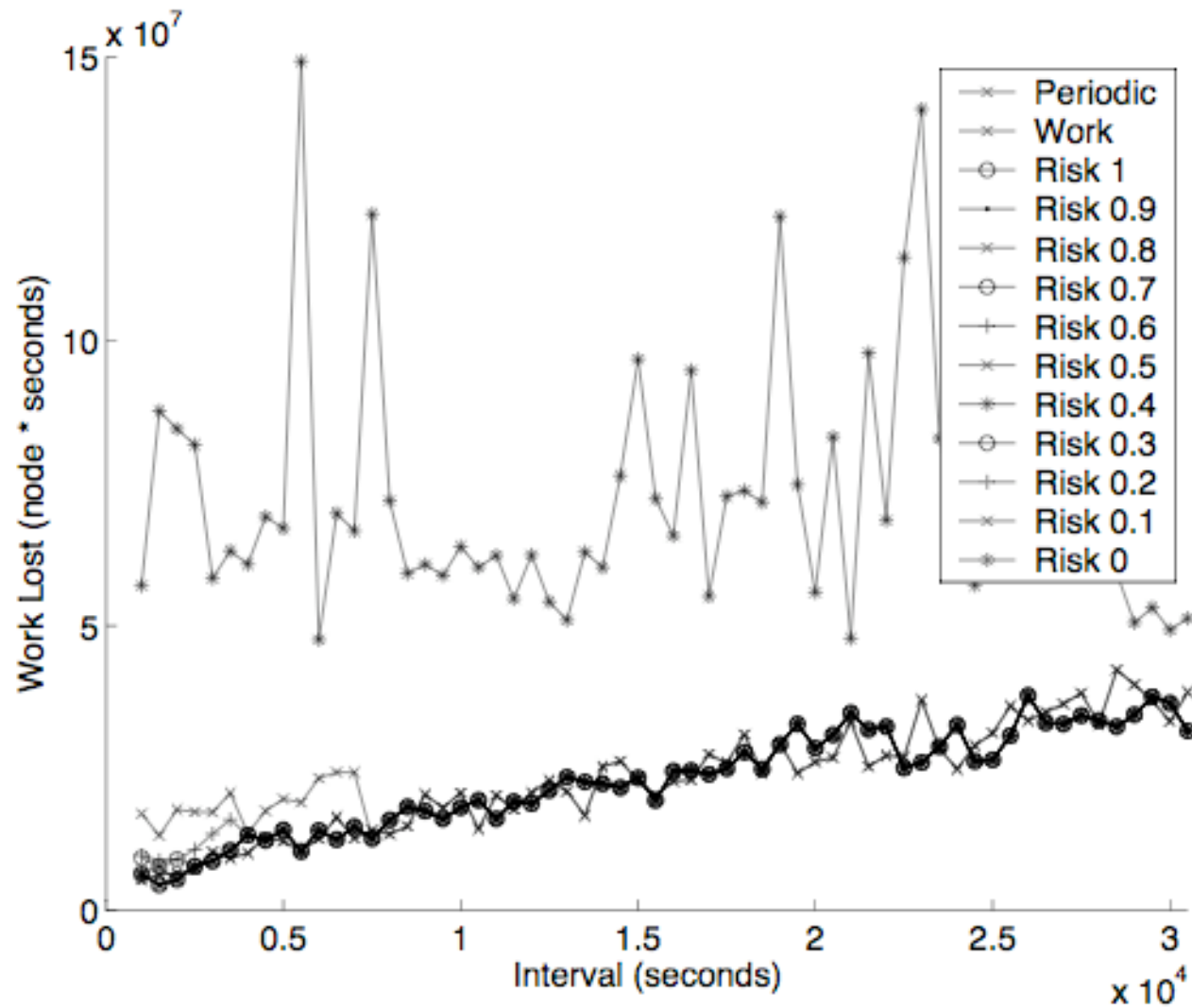


Results: Flat, SDSC, C=3600



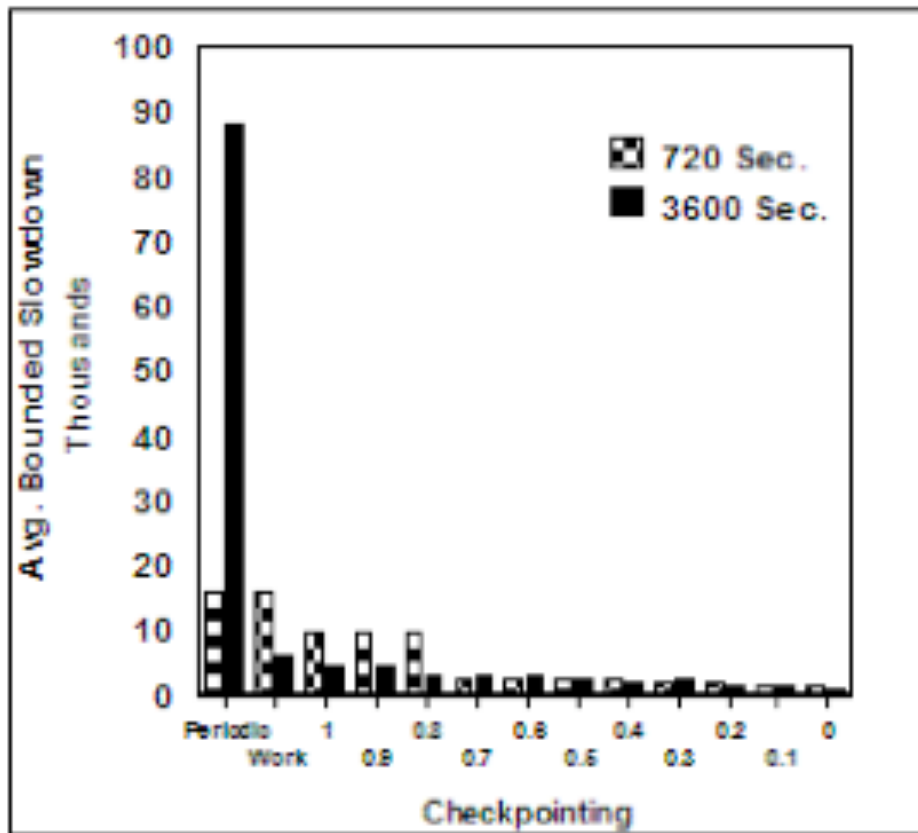


Results: Torus, SDSC, C=720



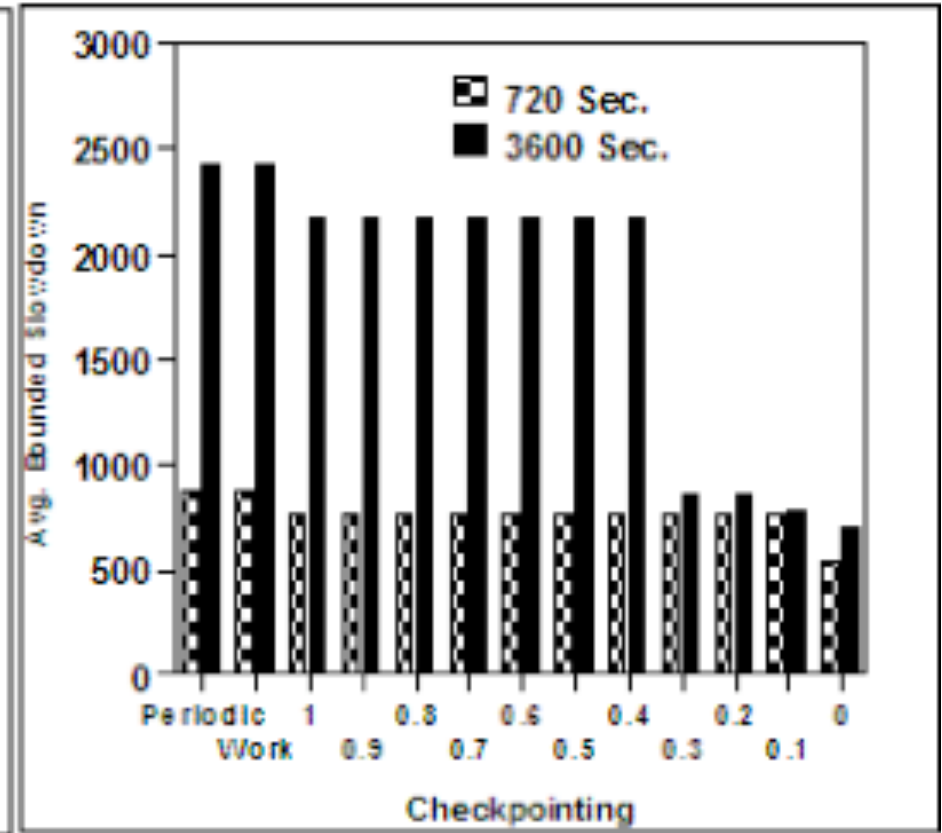


Results: Flat, SDSC



(a)

I = 1000 seconds

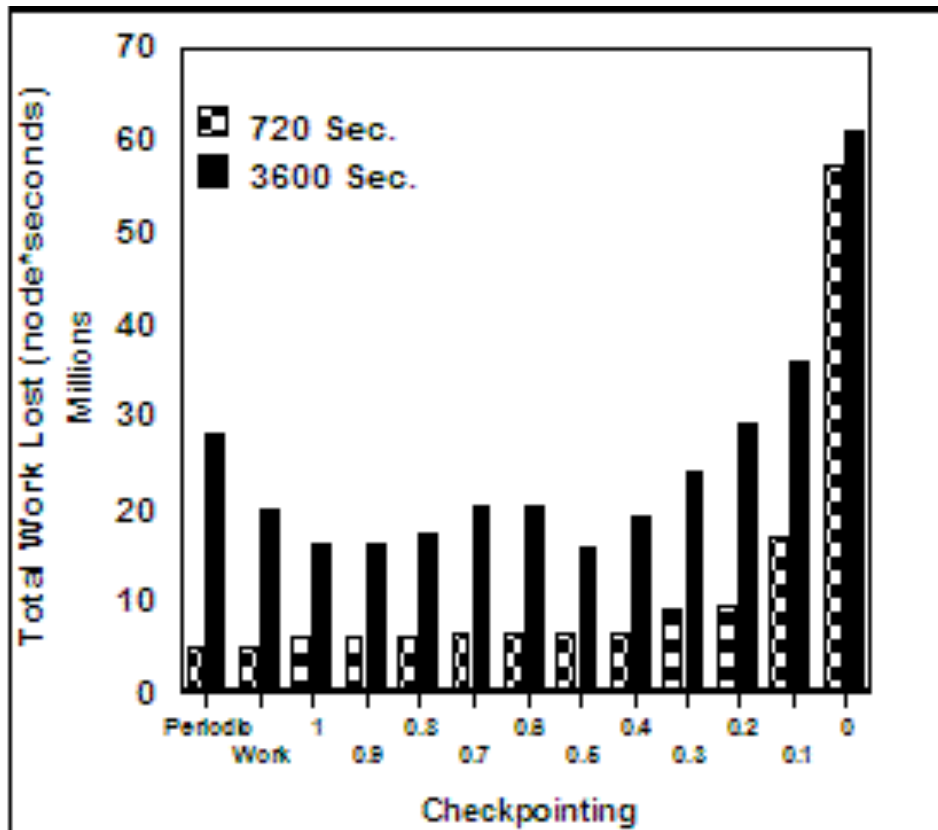


(b)

I = 10000 seconds

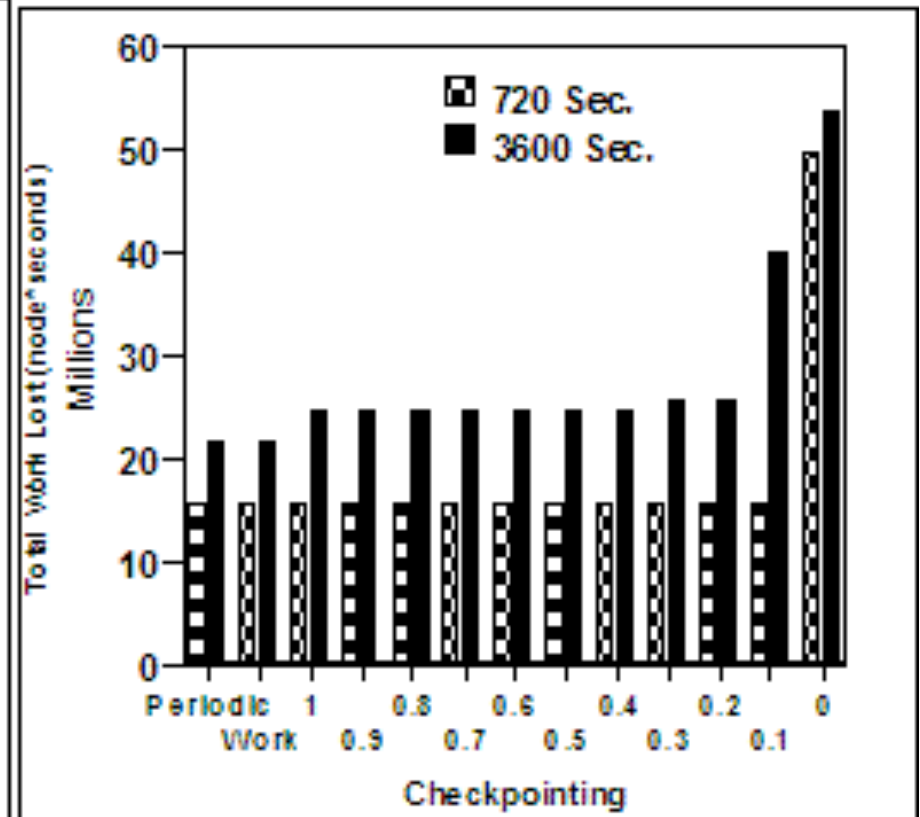


Results: Flat, SDSC



(a)

$I = 1000$ seconds



(b)

$I = 10000$ seconds



Salient Points

- Hell is paved with good intentions
 - Naïve checkpointing can do more damage than good
- A little goes a long way
 - Predicting 10% of failures is enough



Contributions

- Describes a cooperative checkpointing simulator
- Presents results using real workloads and failure logs
- Characterizes the importance of topology
- Shows that simple heuristics can improve system level metrics
- Demonstrates that forecasting with 10% accuracy is sufficient