# Stride Scheduling

Robert Grimm
New York University

# The Three Questions

- What is the problem?
- What is new or different?
- What are the contributions and limitations?

# Motivation

- Scheduling of scarce computer resources (e.g., CPU)
  - Has major impact on throughput and response time
  - Should be fair (scientific applications)
  - But also needs to adjust rapidly (interactive applications)
- Priority-based schemes
  - Rely on ad-hoc assignment of priorities
  - Are poorly understood
  - Do not provide encapsulation, modularity

# A Probabilistic Solution: Lottery Scheduling

# Lottery Scheduling

- Provides a randomized mechanism
  - Not suitable for (hard) real-time systems
- Provides control over relative execution rates
- Can be implemented efficiently
- Supports modular resource management

# The Basic Ingredients

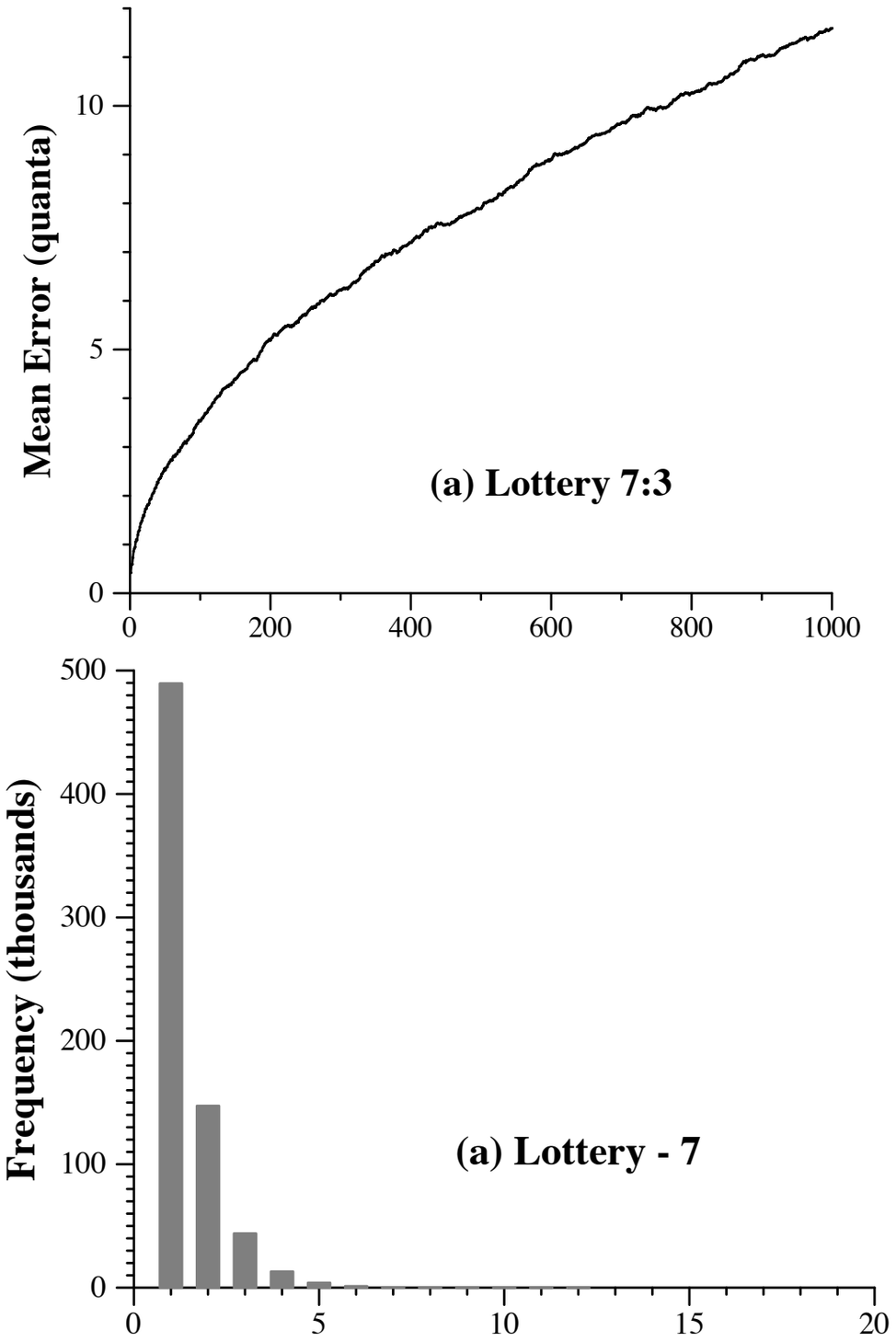
- Tickets
  - Abstract, relative, and uniform resource rights
- Lotteries
  - Probabilistically fair selection of next resource holder
  - Throughput proportional to client's ticket allocation
    - Binomial distribution, accuracy improves with $\sqrt{n}$
    - Average response time inversely proportional to client's ticket allocation

# Fun with Lottery Tickets

- Ticket transfers
  - Useful for RPC-based systems
  - Avoid priority inversion problem
- Ticket inflation
  - Provides alternative to transfers (no communication!)
  - Needs to be avoided/contained in general
- Ticket currencies
  - Support flexible naming, sharing, and protecting of resource rights
- Compensation tickets
  - Make up for underutilization

# Wait a Minute!

- Lottery scheduling is probabilistic
- Mean error slows down, but still grows
- Fairly high distribution of response times



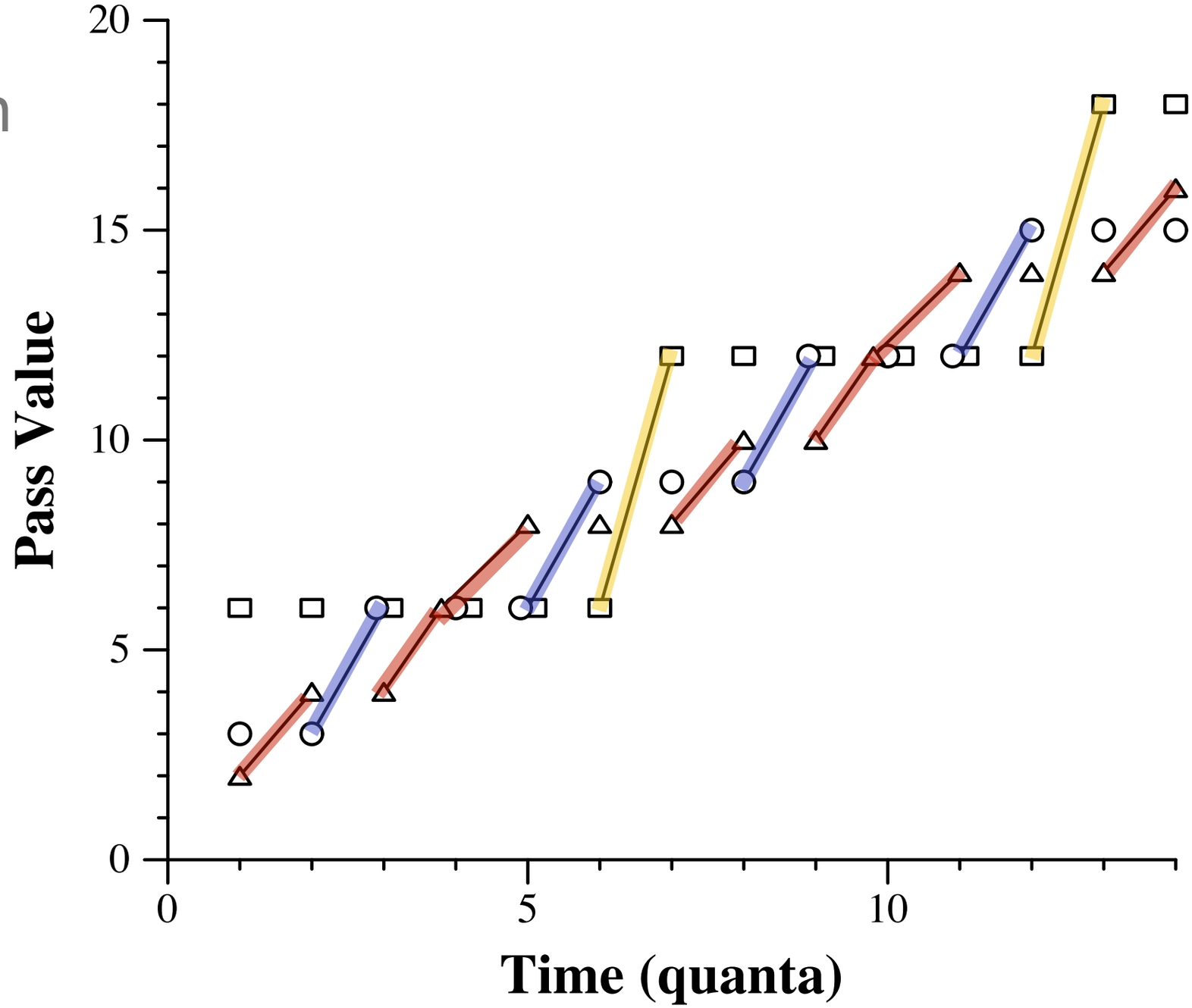
# A Deterministic Solution: Stride Scheduling

# The Basic Ingredients

- Tickets
  - Abstract, relative, and uniform resource rights
- Strides (stride = stride1 / tickets)
  - Intervals between selections
- Passes (pass += stride)
  - Virtual time index for next selection
    - Clients with smallest pass gets selected

# Making Strides

3:2:1
Allocation



# Dynamic Client Participation

- Need to maintain aggregate information
  - Global tickets: sum of all active clients' tickets
  - Global stride: stride1 / global tickets
  - Global pass: incremented by global stride per quantum
- Need to maintain additional per-client information
  - Remain: client's pass - global pass
    - Added back in when rejoining the system

# Dynamic Ticket Modification

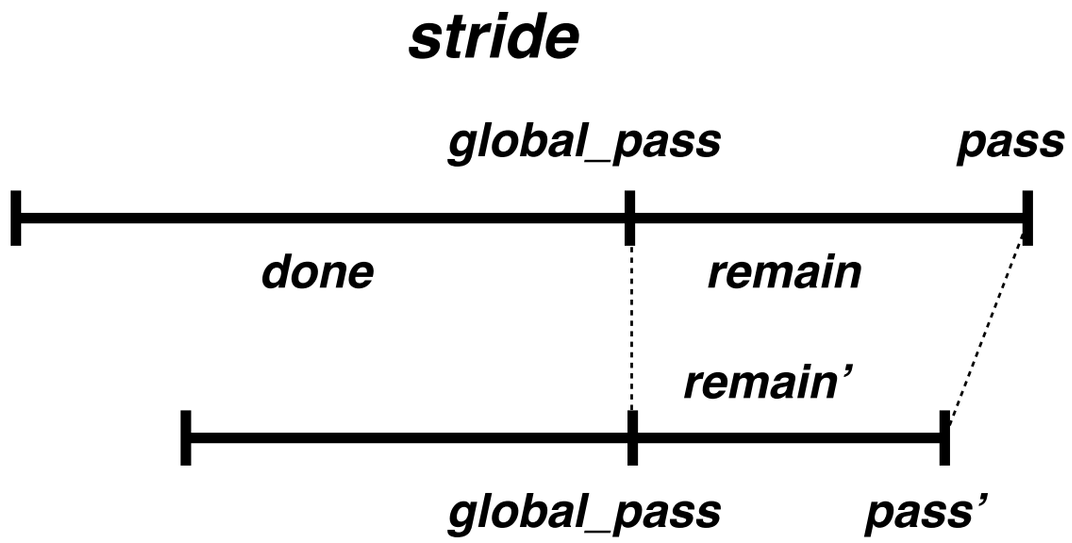



Figure 4: **Allocation Change.** Modifying a client's allocation from *tickets* to $tickets'$ requires only a constant-time recomputation of its *stride* and *pass*. The new $stride'$ is inversely proportional to $tickets'$. The new $pass'$ is determined by scaling *remain*, the remaining portion of the the current *stride*, by $stride' / stride$.

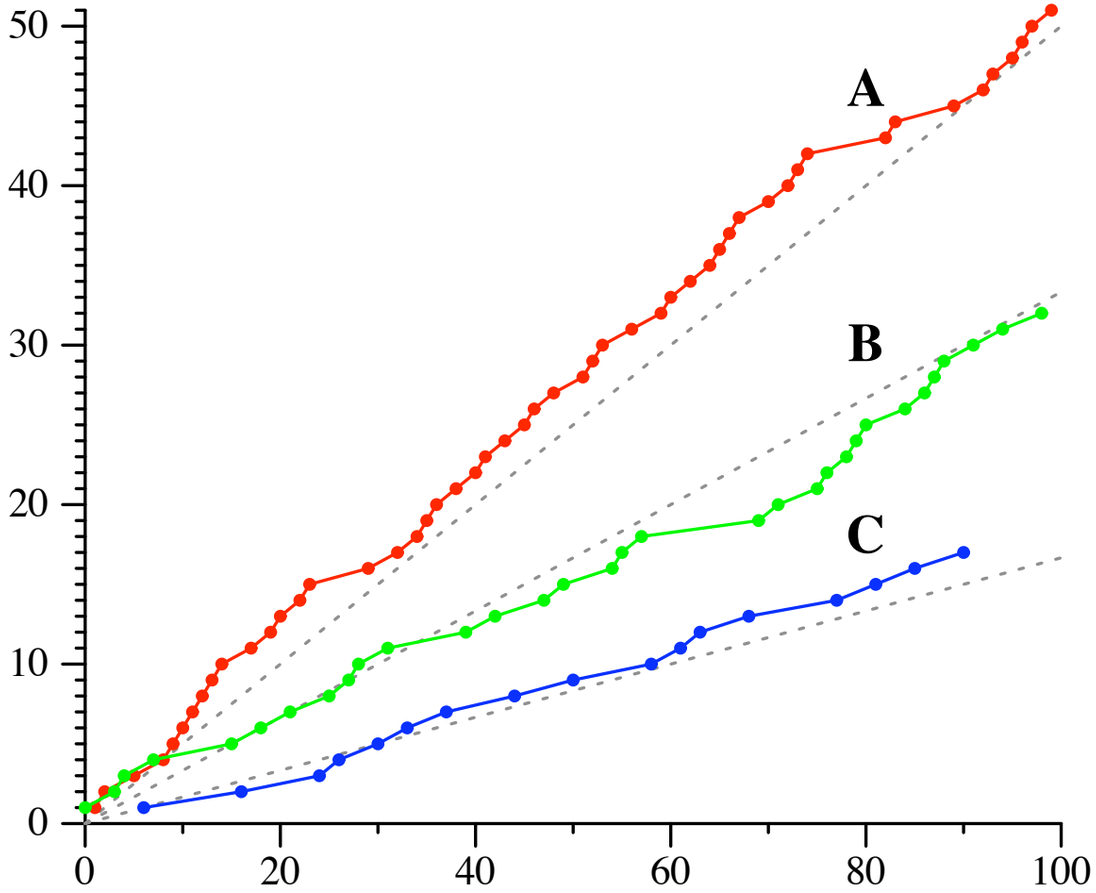
# One More Thing

- Throughput error
  - Bounded to single quantum between any two clients
  - But absolute error may still be O(# clients)
- Approach: combine clients into groups
  - Larger ticket allocations, smaller strides
- Algorithm: balanced binary tree of groups
  - Aggregate ticket, stride, and pass values
  - Updates propagated to each of a client’s ancestors

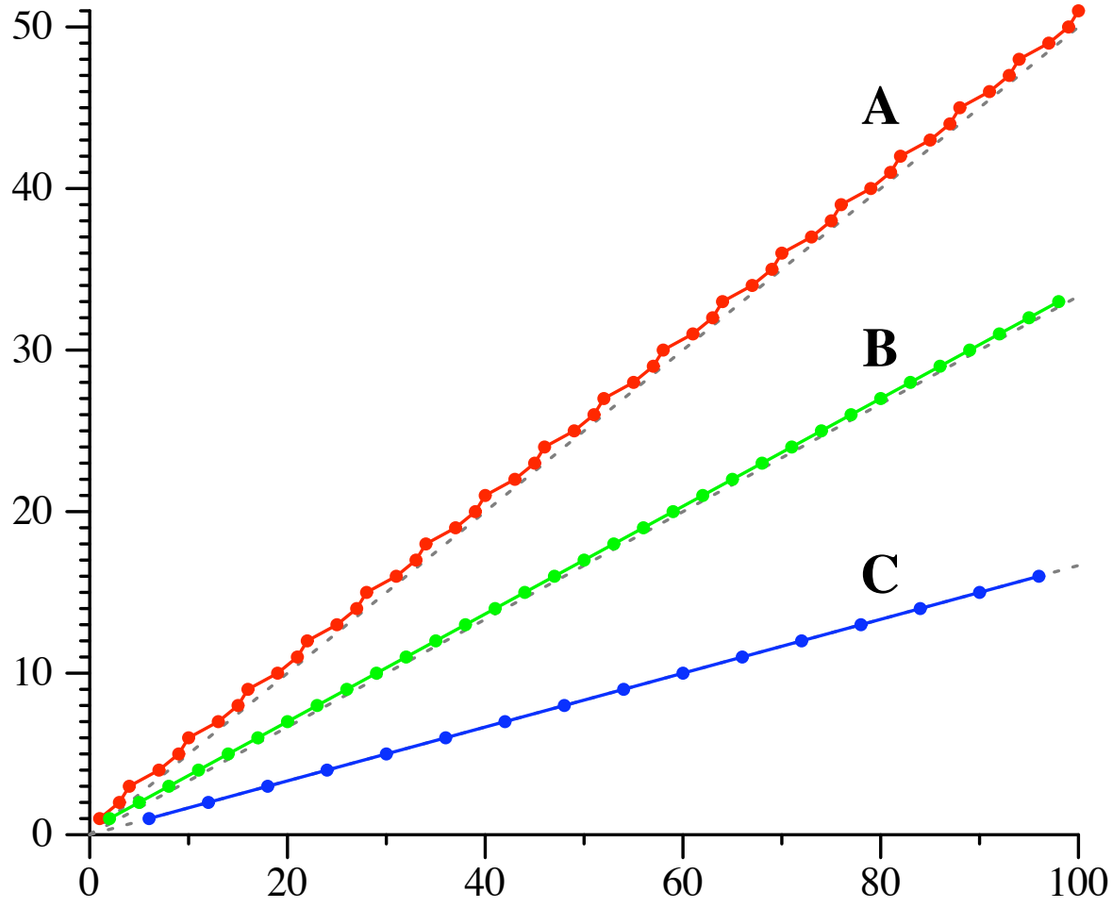
# Simulations

# Lotteries v Strides

## Lotteries

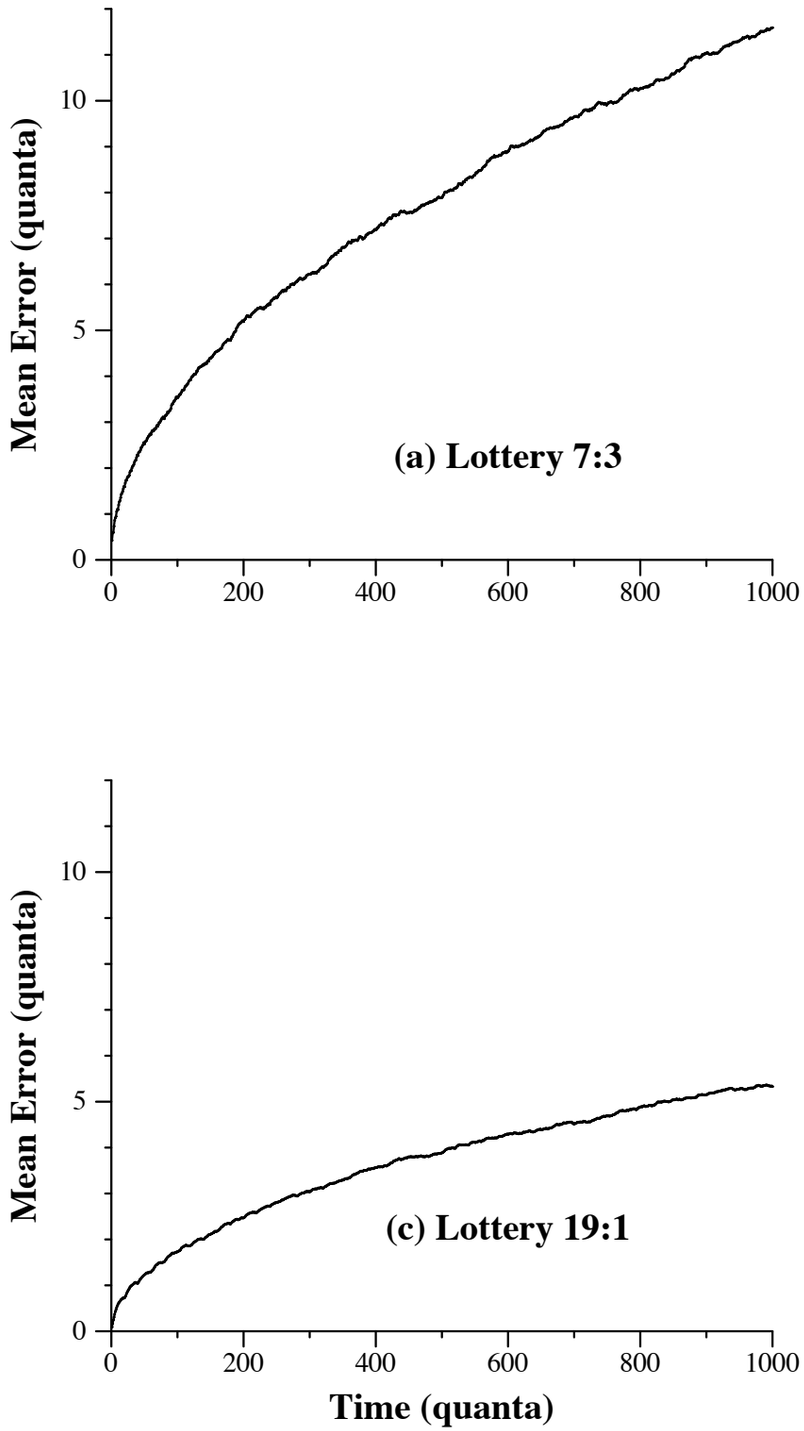


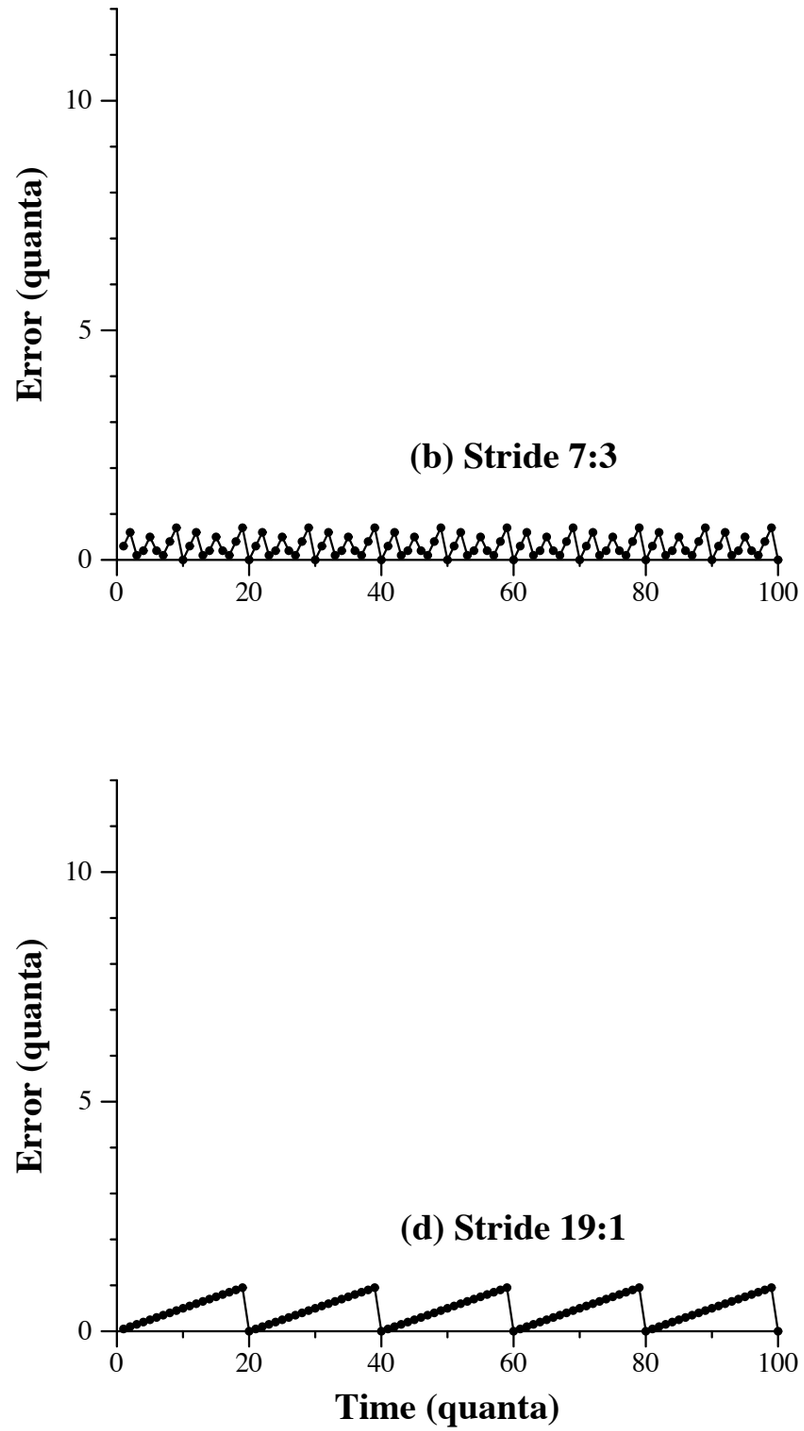

## Strides



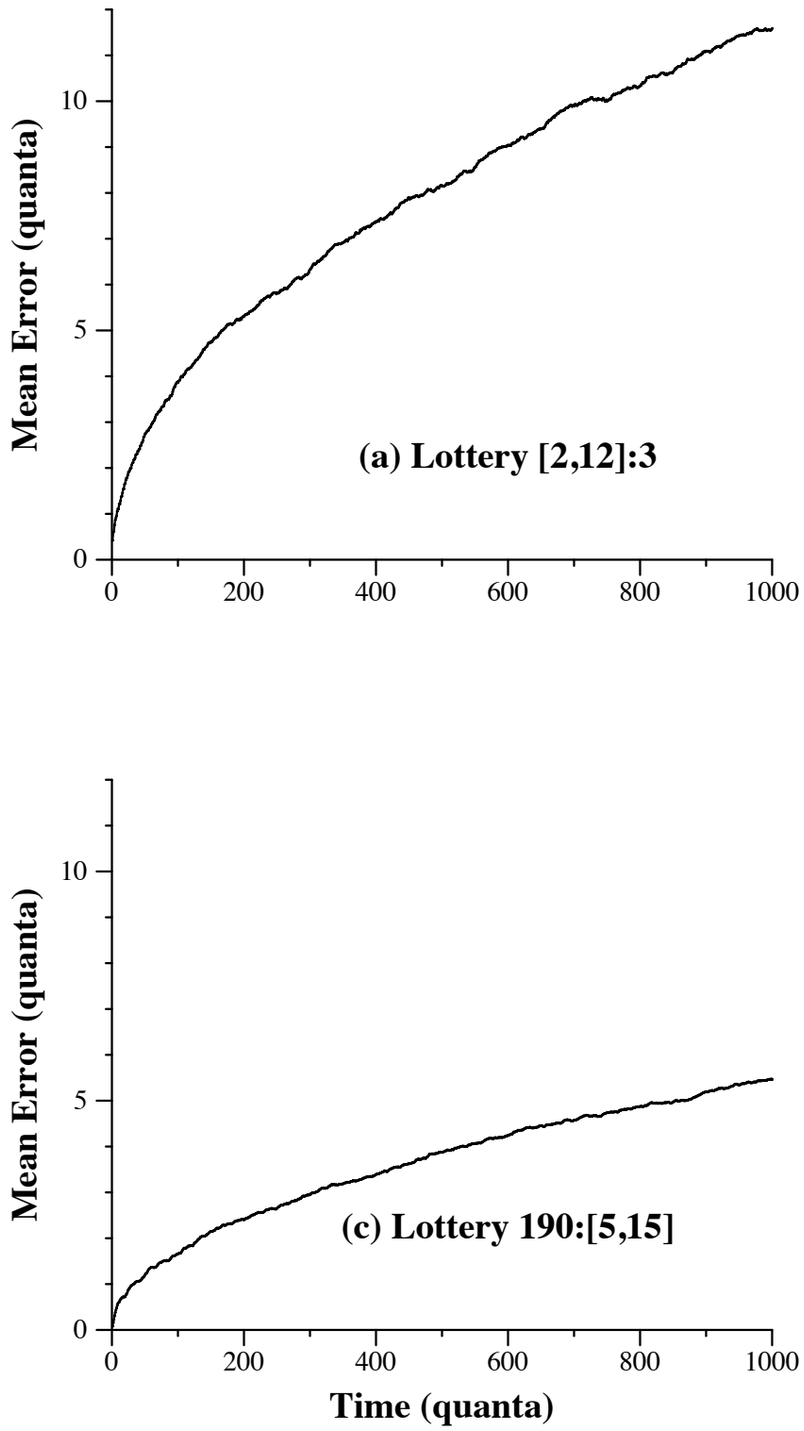

Cumulative Quanta over Time

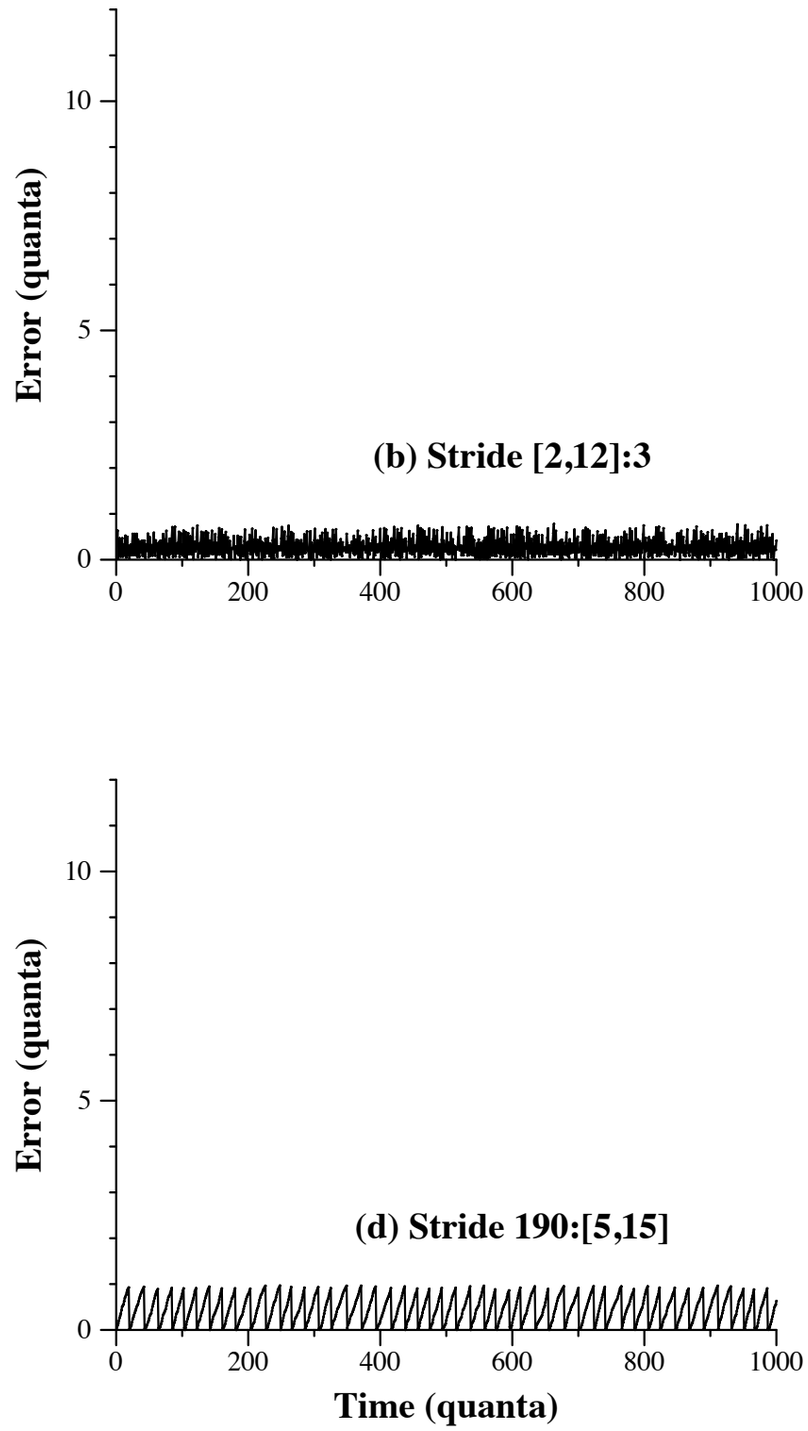
# Throughput Accuracy



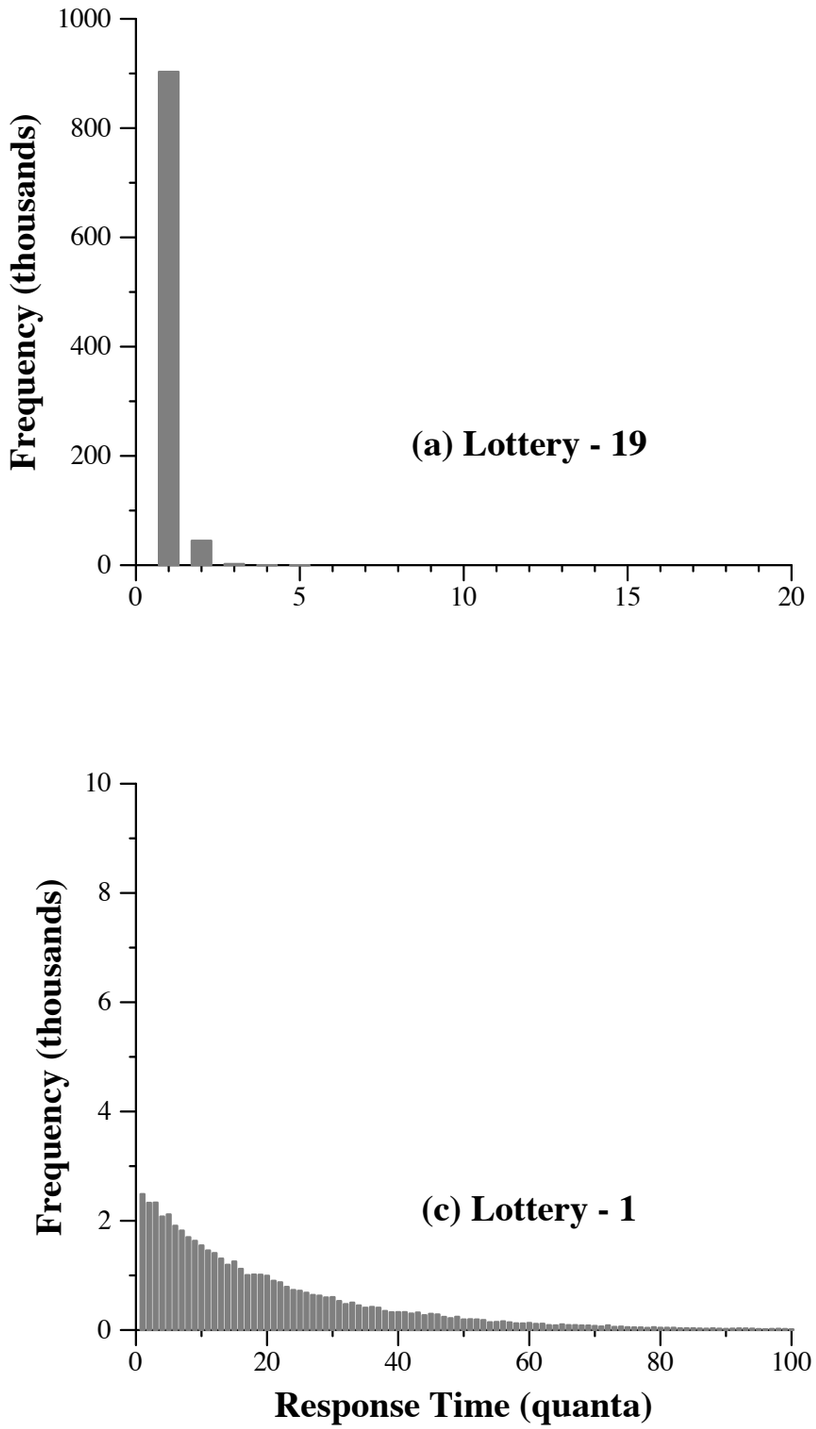



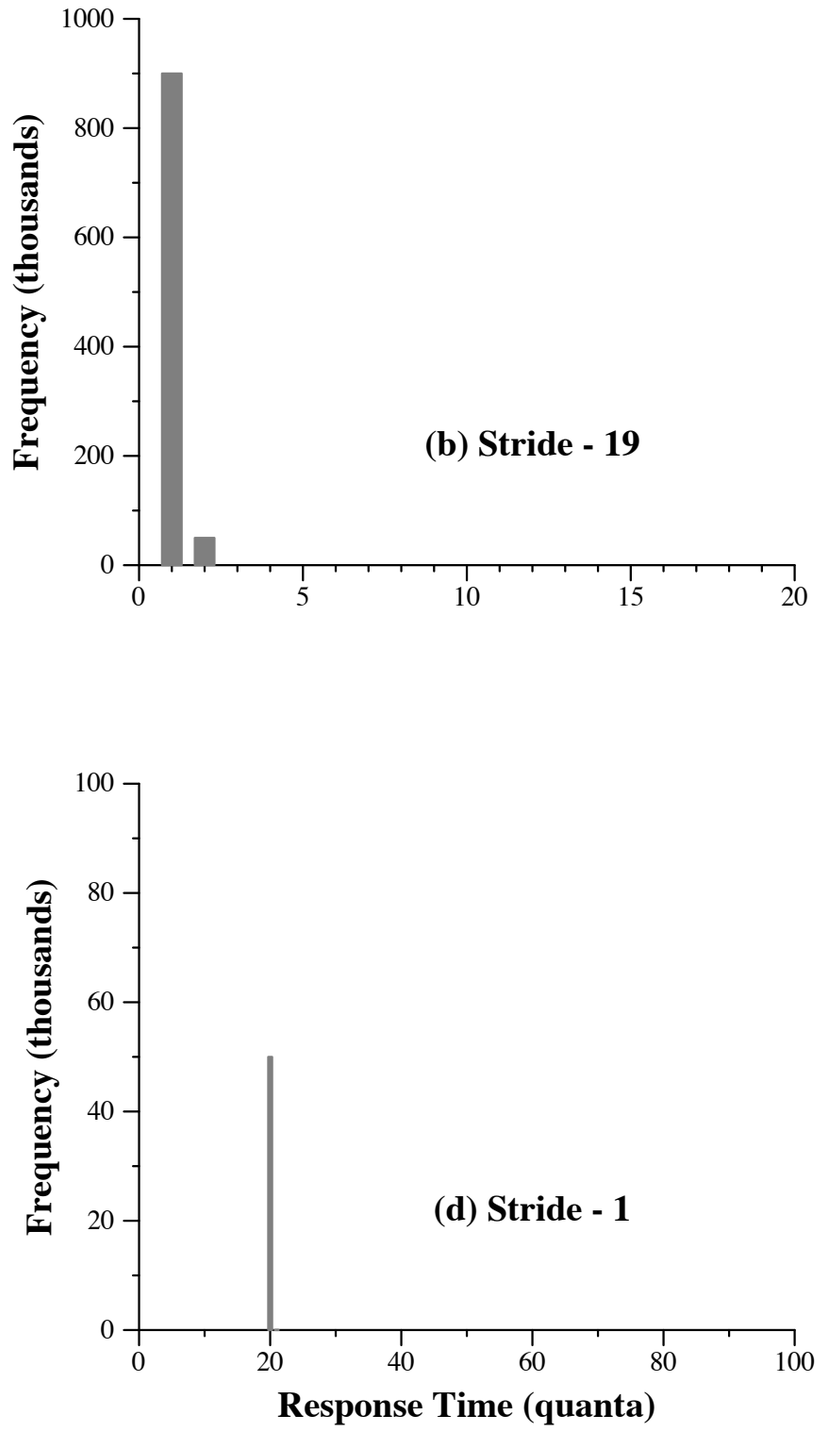
# More Throughput Accuracy



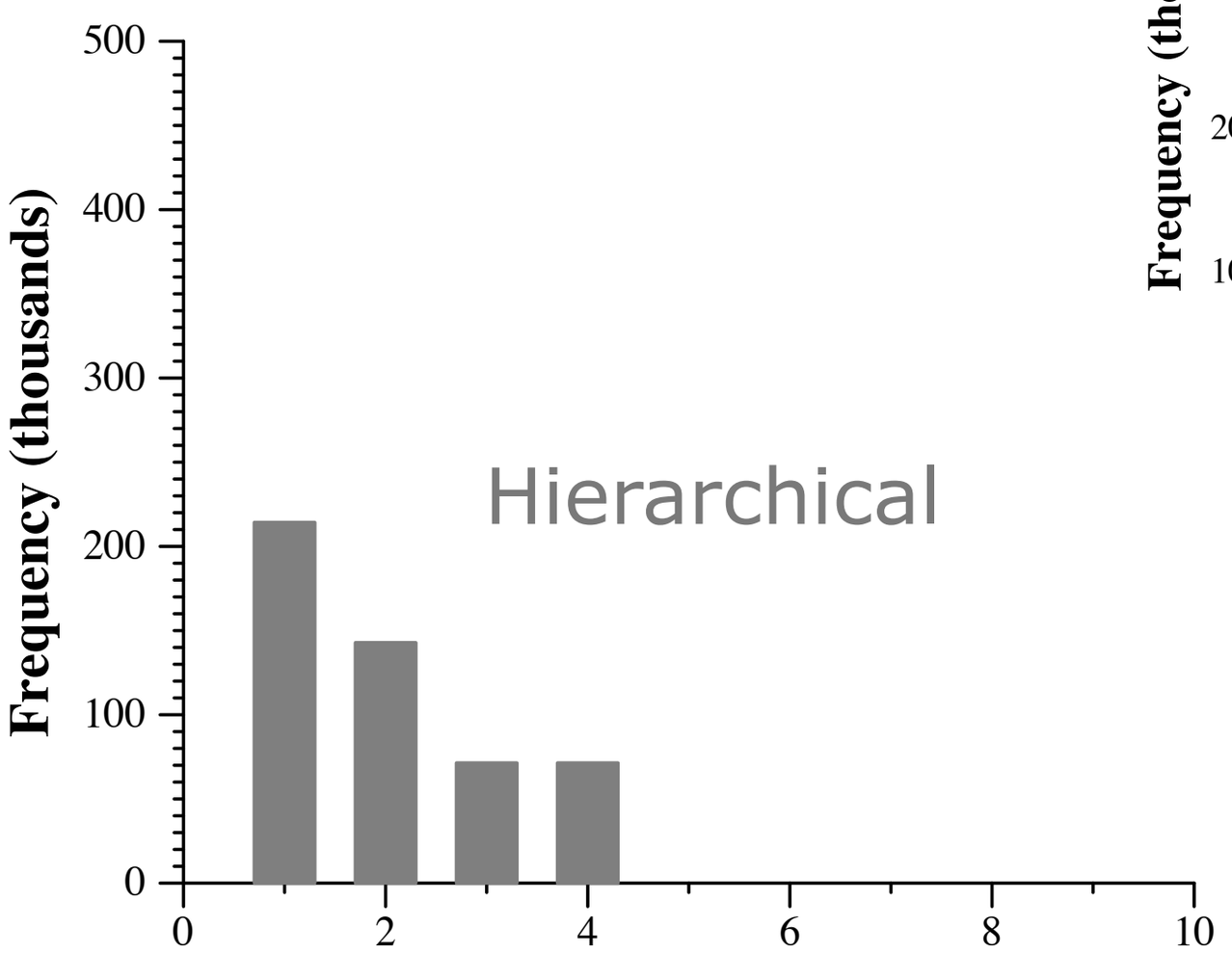



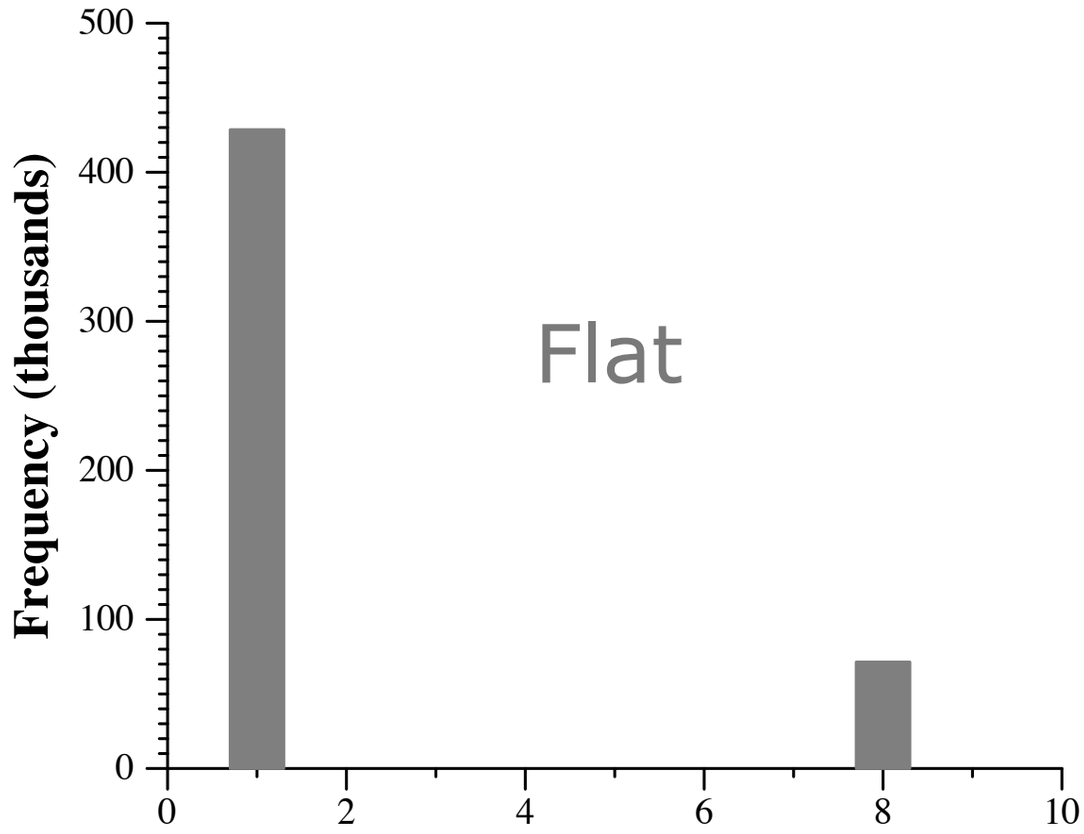
# Response Time Distribution



# Hierarchical Strides



# In the Real World

# CPU Scheduling

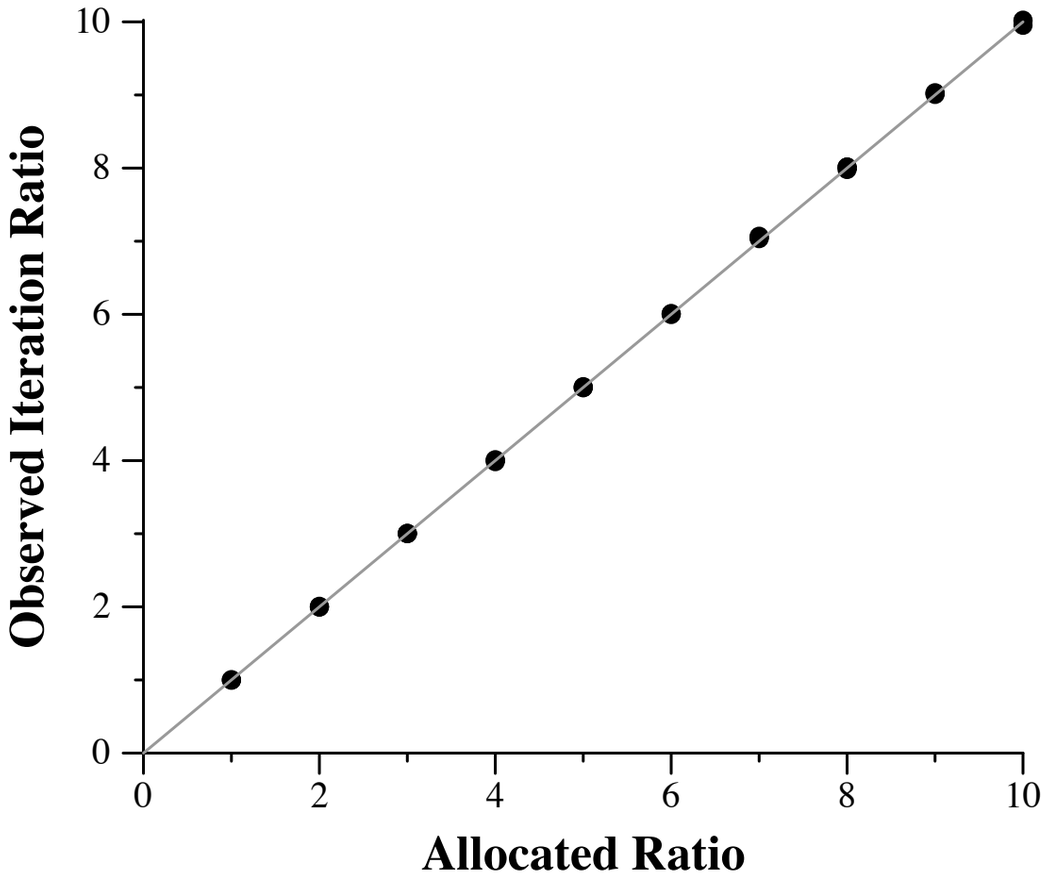



Figure 14: **CPU Rate Accuracy.** For each allocation ratio, the observed iteration ratio is plotted for each of three 30 second runs. The gray line indicates the ideal where the two ratios are identical. The observed ratios are within 1% of the ideal for all data points.

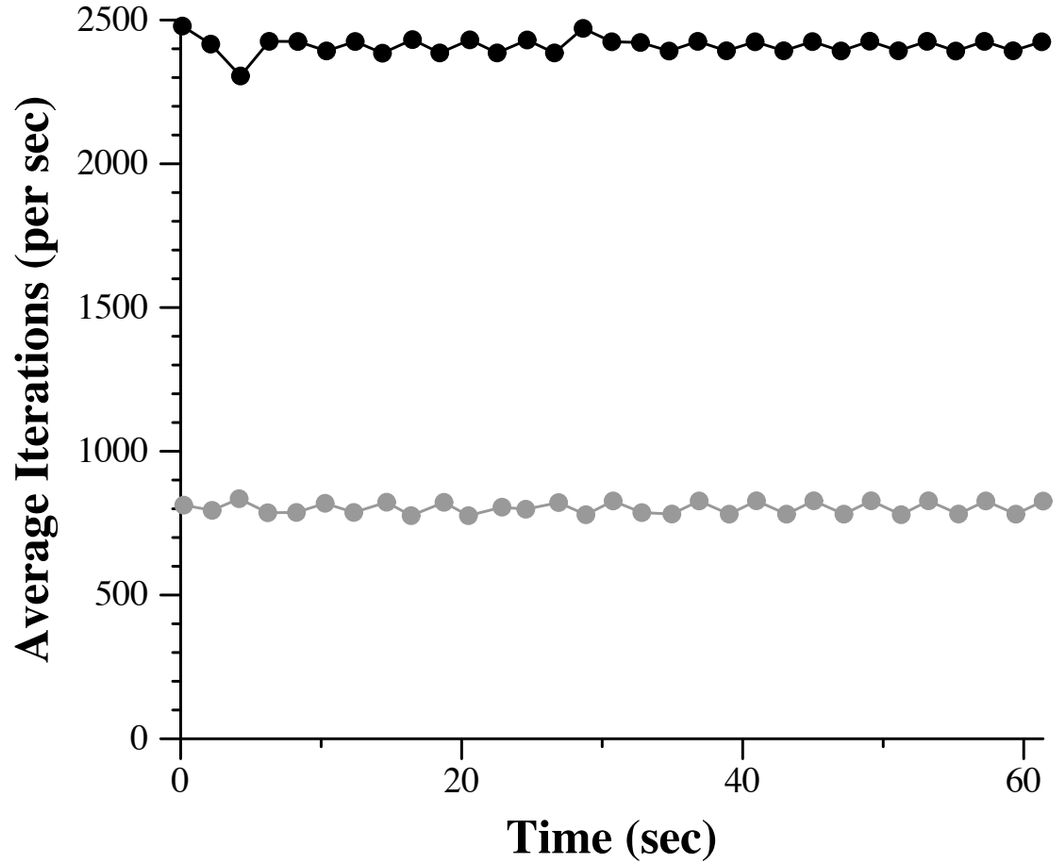



Figure 15: **CPU Fairness Over Time.** Two processes executing the compute-bound `arith` benchmark with a 3 : 1 ticket allocation. Averaged over the entire run, the two processes executed 2409.18 and 802.89 iterations/sec., for an actual ratio of 3.001:1.

# Network Scheduling

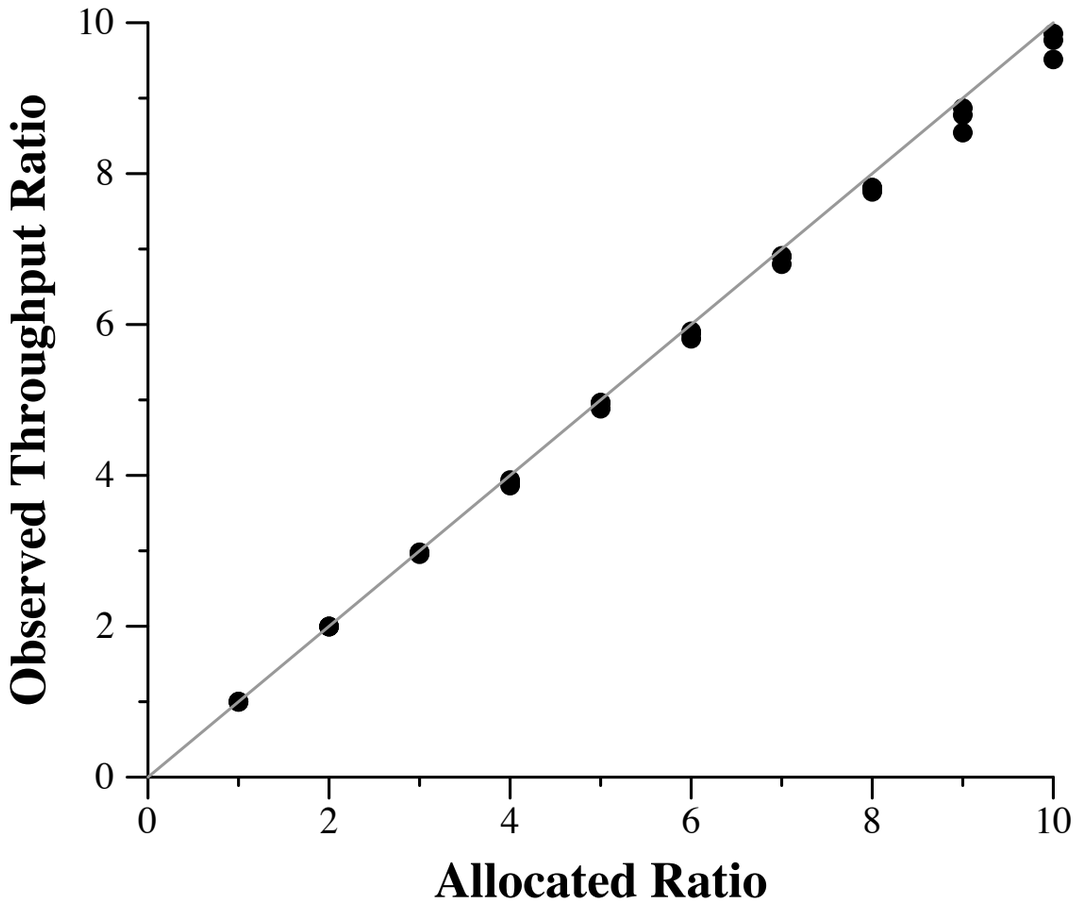



Figure 16: **Ethernet UDP Rate Accuracy.** For each allocation ratio, the observed data transmission ratio is plotted for each of three runs. The gray line indicates the ideal where the two ratios are identical. The observed ratios are within 5% of the ideal for all data points.

# What Do You Think?