1. Context
Stream processing applications
• Multi-processor system
• End-to-end performance requirements
Multiple streams processed concurrently
• User starts and stops streams
• Not all streams completely characterised  → Budget schedulers

2. Problem
Task graphs
• Can be cyclic
• Use blocking writes (wait on sufficient space)
Dataflow is natural model
• Suff. conditions for functional determinism
• Known how to model static-order schedules
• Unknown how to model run-time scheduling

3. Contribution
Dataflow model of an individual application scheduled by run-time schedulers
• Functionally deterministic task graph
• Budget schedulers
Conservative simulation of individual application
• Prerequisite for conservative analysis and synthesis

4. Approach
Task graph
• One-to-one relation with functionally deterministic dataflow graph
Functionally deterministic dataflow
• Temporally monotonic, smaller firing duration cannot lead to later token arrivals

5. Including Effects of Budget Schedulers
Worst-case enabling time
Response time: \( \hat{f}(i) = \hat{e}(i) + x(i) + (P - B) \left( \frac{x(i)}{B} \right) \)
Contribution: \( \hat{f}(i) = \max(\hat{e}(i) + P - B, \hat{f}(i - 1)) + P \frac{x(i)}{B} \)
Improved model
• Latency and rate parameters instead of response time
• Captures multiple executions in a single budget
• Valid for sequences of execution times

6. Accuracy and expressivity
Comparing cycle-true and timed dataflow simulation
Sequence of execution times
Data-dependent production quanta

7. Conclusion
Accurate conservative dataflow model is proposed
• Includes effects of run-time scheduling
• Has monotonic temporal behavior
• No scheduling anomalies in model
Monotonicity (and its generalization: linearity) is basis for my other work that computes budget and buffer sizes satisfying latency and throughput constraints with variable-rate phased dataflow graphs

Scheduler classes:
- Deterministic
  - Latency-rate
  - Budget
Application classes:
- Functionally Deterministic
  - Variable-Rate Phased
  - Cyclo-Static