WOSP/SIPEW 2010

First Joint WOSP/SIPEW International Conference on Performance Engineering

Analytical Modeling of Lock-based Concurrency Control

with arbitrary

Transaction Data Access Patterns

Pierangelo Di Sanzo, Roberto Palmieri, Bruno Ciciani, Francesco Quaglia

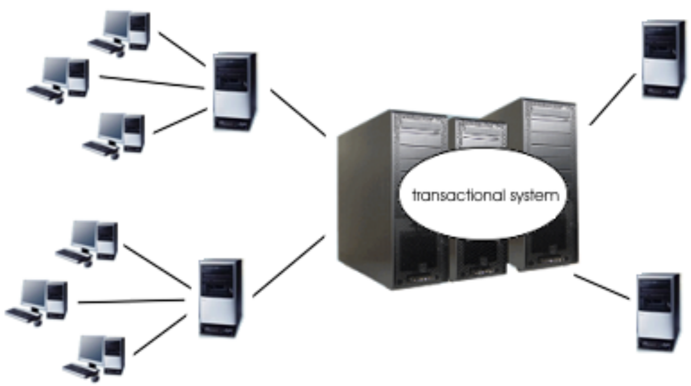
Sapienza, Università di Roma, Italy

Paolo Romano

INESC-ID, Lisbon, Portugal

Transactional systems

- Transactional systems (TS) are fundamental building blocks for a lot of modern IT systems
- E.g., database systems, transactional memories, transactional file systems,…
- Basically this is due to their ability to ensure so-called ACID properties (Atomicity, Consistency, Isolation, Durability) in presence of concurrency
- Given the relevance of TS, accurate methodologies/techniques to study their performance are mandatory



CC component

- Transactional Systems are formed by various subsystems/ components, closely working with each other
- Concurrency Control (CC) is the main TS proper component
- As for evaluation aspects, it is typically hard to isolate/capture CC overheads while the TS is operative
- It becomes extremely relevant to characterize CC via a ah-hoc models, easy to be integrated with models capturing the effects of other subsystems (e.g. buffer-pool models)
- The performance of CC is influenced by:
  - hardware resources contention
  - data contention

Data contention: Workloads

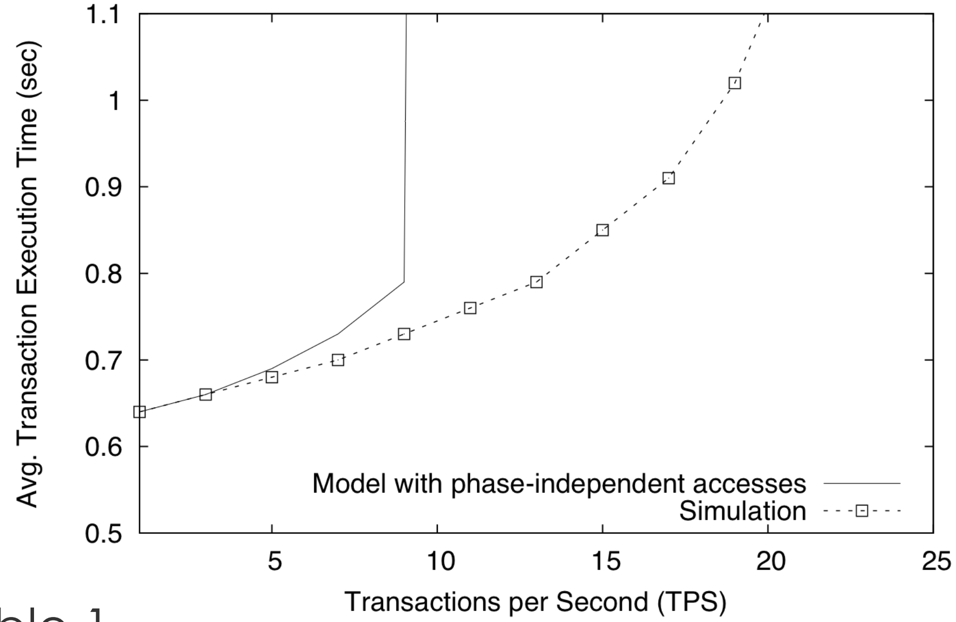
- Performance of transactional systems are impacted by the type of workload
- In most of cases, applications follow a set of specific patterns to access data
- Neglecting some specific workload features could bring to non-realistic modelling/evaluation results
- One target is therefore the design of performance models capturing/expressing workloads representative of actual application patterns (transactional classes, sequence of accesses,…)

Existing analytical models for CC / 1

- Workload description is an input for these models
- Main lacks of existing analytical models:
  - Data Access Pattern
    - Neglect the specific data item accessed within each execution phase of a transaction but consider, for each phase, the same data access distribution over the whole data item set
  - High contention
    - Limit data contention on each data item
    - Not capture phenomena (e.g. lock-queuing bursts) due to high concurrency

Existing analytical models for CC / 2

- Example of divergence of the response time between :
  - simulation
  - an analytical model not capturing sequences (phases) of data accesses and high contention phenomena



- Workload details:
  - 1st to 5th operation executed on Table 1
  - 6th to 10th operation executed on Table 2
  - 11th to 15th operation executed on Table 3
  - 16th to 20th operation executed on Table 4

Contributions

- Analytical model of lock-based Concurrency Control
- Two main innovative features:

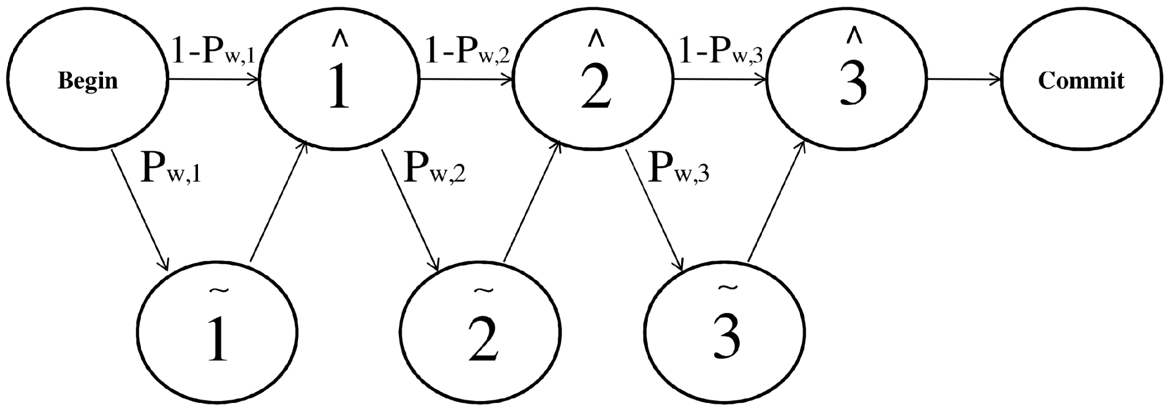
- Accurate modeling of Transactions' data access pattern
- Ability to express/represent realistic system workloads

- Ability to capture lock-queuing phenomena in situations with heavy data contention among transactions

Model Description

- Strong Strict Two-Phase Locking Concurrency Control
- Each transaction is composed by M states from Begin to Commit
- Transaction executes one operation (read or write) per state
- Due to data contention, the execution could enter in a wait state, that is left when the locker transaction commits/aborts

Model of transaction's execution



Data Access Pattern

- The transaction access pattern is modeled by *[I x M]* matrix called *A* where:
  - *I* represents number of data items
  - *M* represents the phases of transaction's evolution
- Each cell $A_{i,k}$ of access matrix represents the probability that the k^{th} operation of transaction accesses the i^{th} data item
- The typology of transaction's operations is modeled by vector *W*
- $|W| = M$
- W_k represents the probability that k^{th} operation is a write

Access Matrix

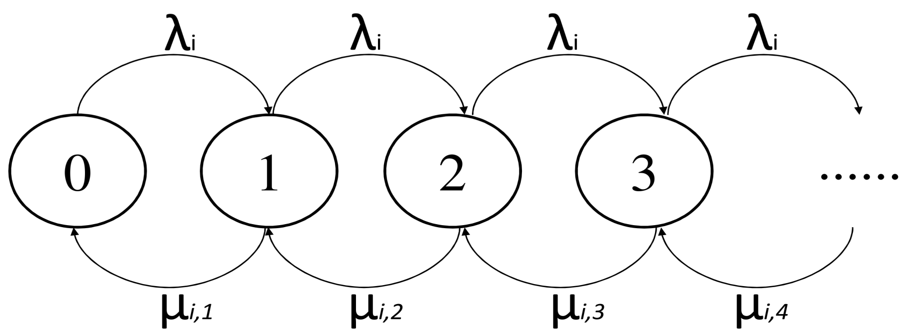
0,5	0
0,5	0
0	1
0	0

Operations Vector

0
1

High data contention

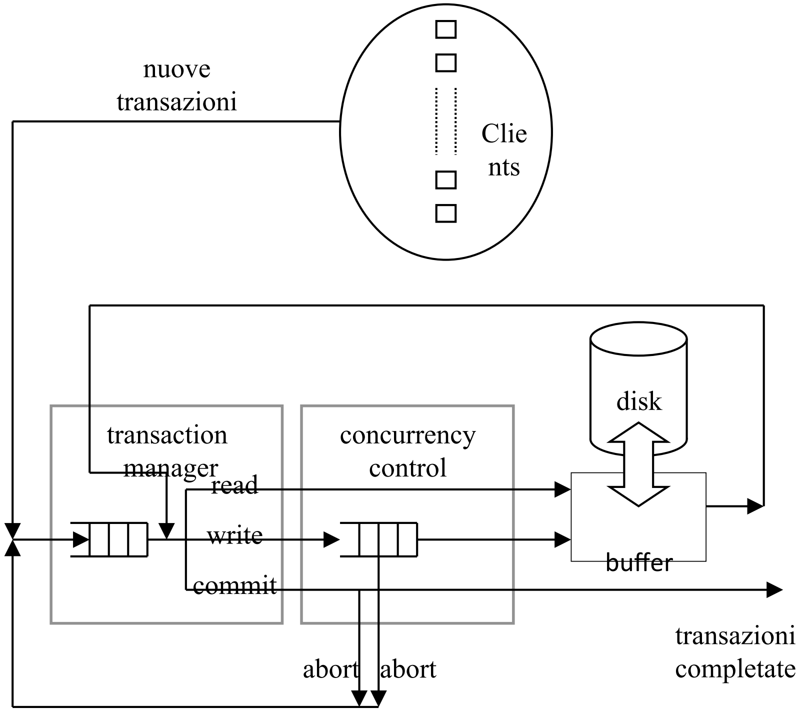
- To cope with the evaluation of data contention, we have modeled the arrival-departure requests for each single data item *i* as a birth-death process with:
  - average arrival rate λ_i
  - average service rate $\mu_{i,j,}$ where *j* corresponds to the number of standing requests for data item *i* in the corresponding state of the Markov chain
- We evaluated $\mu_{i,j}$ on the basis of interleaving of read and write requests observed in state j (i.e. considering an approximate probability distribution of the number of top standing reads)



Validation & Simulator

- We validate the model comparing numerical results with output of discrete event simulator developed using the C programming language
- Validation is performed via several scenarios:
  - Synthetic workloads
  - Workloads derived by abstracting the main features of the transaction profiles specified by the TPC-C benchmark

"DBSimulator"

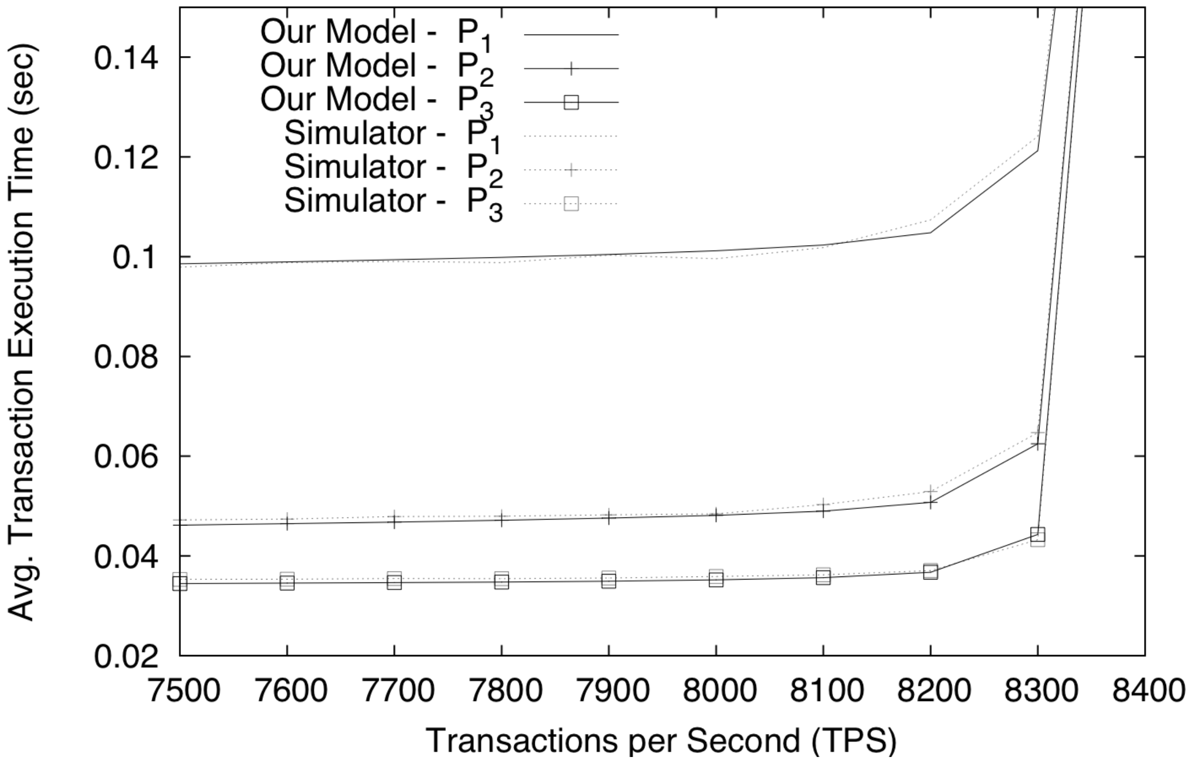


Description of Synthetic Workload / 1

- Data items are grouped in 5 contiguous sets (logically equivalent to, e.g., database tables)
- 20% write operations
- Workload 1:
  - Uniform distribution on each set of data items
  - Three transactions profiles:
    - Profile 1: 4 operations in each data item set (20 total operations)
    - Profile 2: 4 operations on set 1 and after other 4 on set 4 (8 total operations)
    - Profile 3: 4 operations on set 4 and after other 4 on set 5 (8 total operations)

Validation: Synthetic Workload 1

- Numerical results of model follow the output of simulator also near the saturation point of system. This is the effect of capability's model to capture high contention scenarios

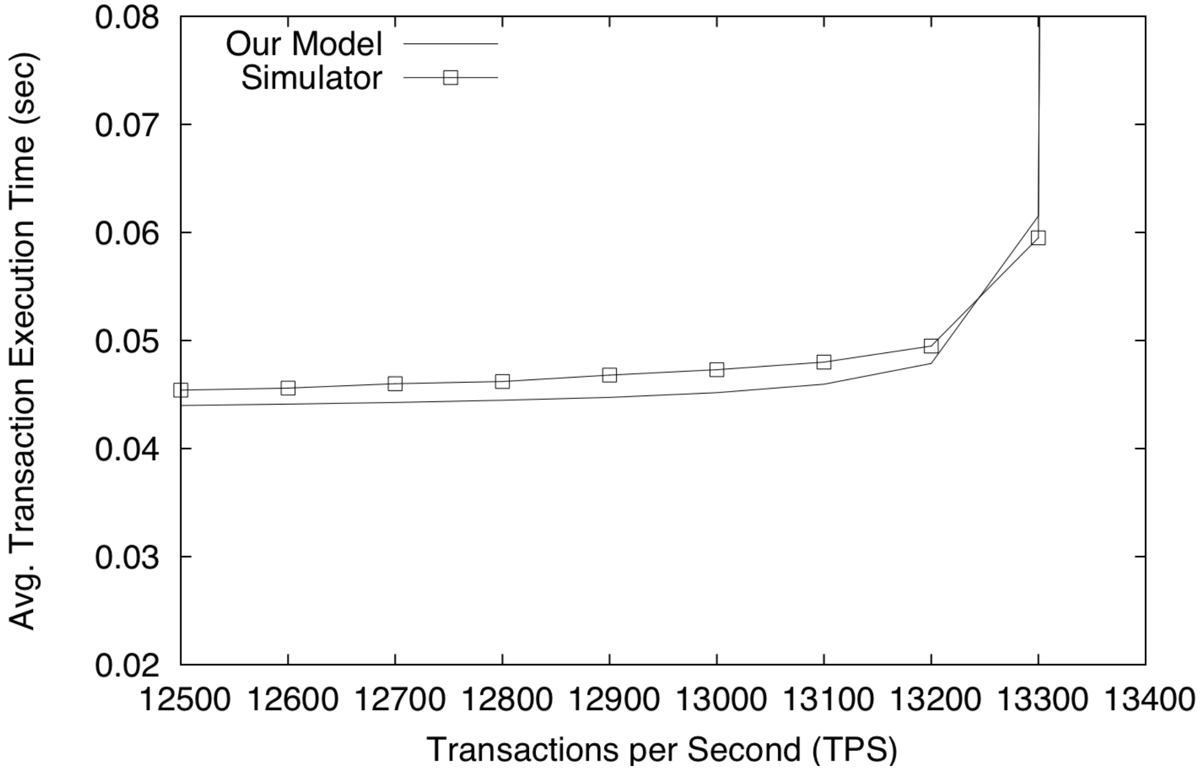


Description of Synthetic Workload / 2

- Workload 2:
  - Represents a “stress” case
  - 15 operations for each transactional profile
  - Two transactions profiles with symmetric data access pattern:
    - Profile 1: 3 accesses to each set S_i starting from S_1 and then sequentially moving according to increasing set indexes
    - Profile 2: 3 accesses to each set starting from S_5 and then moving to the other sets according to a (reverse) decreasing order of the set indexes

Validation: Synthetic Workload 2

- In such a configuration, items in the sets with extreme indexes (i.e. index 1 and index 5) experience lock holding times with high variance across the two transaction profiles



Description of TPC-C benchmark

- Workload reflecting relevant features of a standard benchmark for transactional systems (TPC-C)
- Four main transaction profiles (P_i)

Database configuration

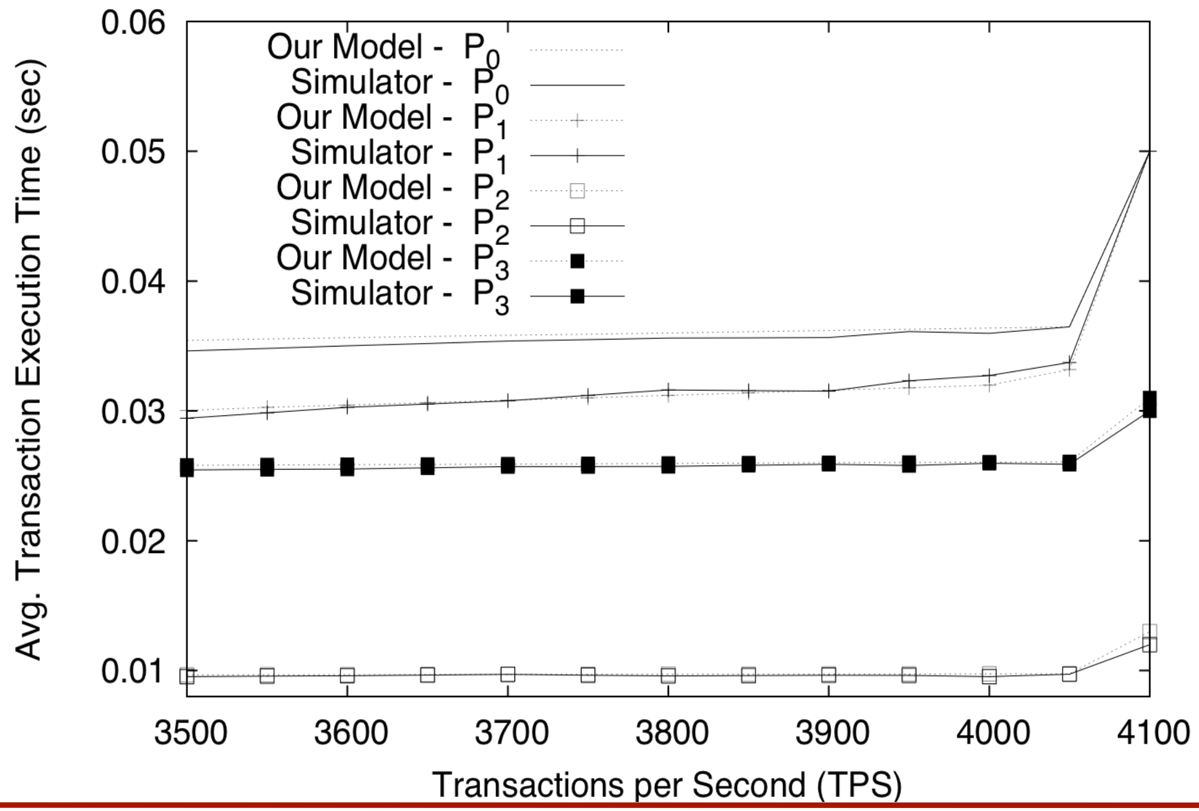
Table Name	# Items	
WAREHOUSE	500	tb0
DISTRICT	1000	tb1
CUSTOMER	15000	tb2
STOCK	500000	tb3
ITEM	100000	tb4
ORDER	1000	tb5
NEW-ORDER	1000	tb6
ORDER-LINE	1000	tb7
HISTORY	1000	tb8

Access Matrix

Phase	P_0	P_1	P_2	P_3
0	(R),tb0	(R),tb0	(R),tb2	(R),tb6
1	(R),tb1	(R),tb1	(R),tb5	(W),tb6
2	(W),tb1	(R),tb2	(R),tb7	(R),tb5
3	(R),tb2	(W),tb0		(W),tb5
4	(W),tb5	(W),tb1		(R),tb7
5	(W),tb6	(W),tb2		(W),tb7
6	(R),tb4	(W),tb8		(R),tb2
7	(R),tb3			(W),tb2
8	(W),tb3			
9	(W),tb6			

Validation: TPC-C Workload

- By the results it can be observed that our model well fits the simulation output



Assessments and future works

- Apply data access pattern methodology in other no lock-based concurrency control
- Propose innovative framework to capture dynamically profiles of transactions and auto-compose data access matrix

Thanks