Verifying Parameterized taDOM+ Lock Managers

Antti Siirtola,
University of Oulu, Finland

Michal Valenta,
Czech Technical University in Prague

taDOM Protocols

- provide transactional access to XML database
- implement DOM interface that provides tree-based access to XML data
- lock-based
- taDOM2, taDOM2+ for DOM level 2, taDOM3, taDOM3+ for DOM level 3
- We consider the lock managers of taDOM + protocols.

taDOM+ Protocol Example

- multiple access to library database
- node lock modes
  - SR – subtree read
  - SX – subtree write
  - IR – intension to read in a subtree
  - IX – intension to read or write in a subtree
  - SRIX – subtree read and intension to write in a subtree
  - SR and IR require IR on the parent
  - other lock modes require IX on the parent

taDOM+ Protocol Example

- compatibility matrix

	IR	SR	IX	SRIX	SX
IR	+	+	+	+	
SR	+	+			
IX	+		+		
SRIX	+				
SX					

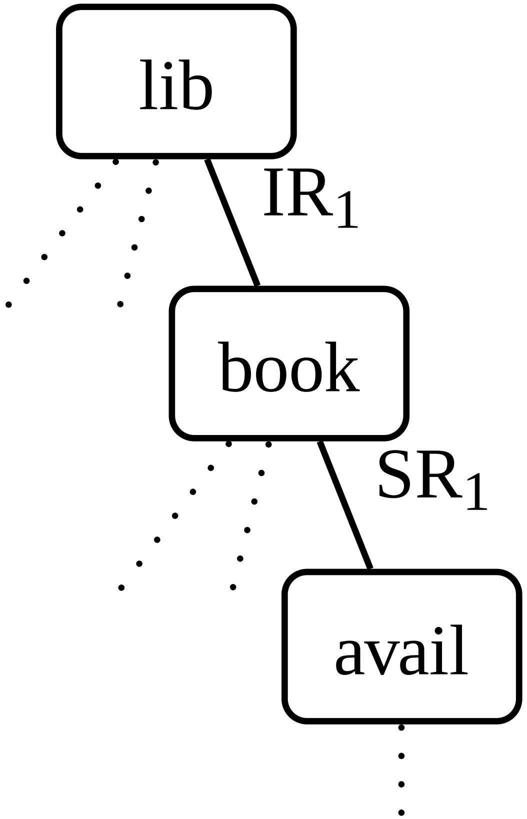
taDOM+ Protocol Example

- conversion matrix

	IR	SR	IX	SRIX	SX
IR	IR	SR	IX	SRIX	SX
SR	SR	SR	SRIX	SRIX	SX
IX	IX	SRIX	IX	SRIX	SX
SRIX	SRIX	SRIX	SRIX	SRIX	SX
SX	SX	SX	SX	SX	SX

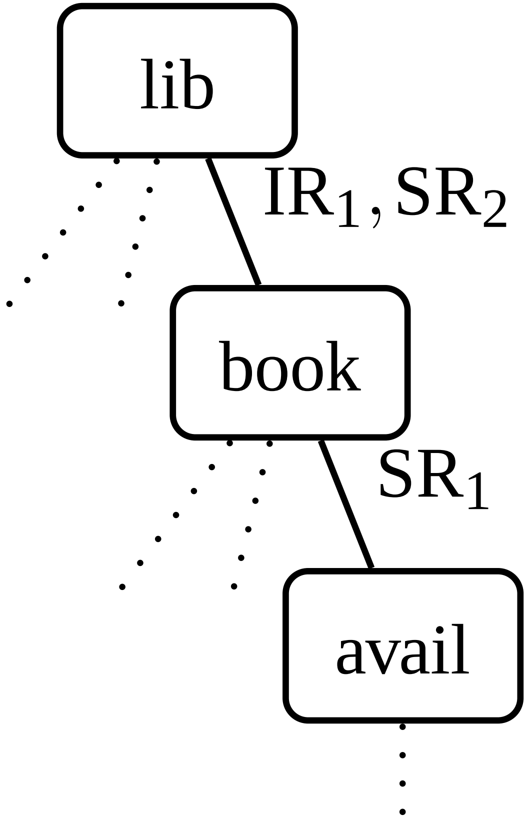
taDOM+ Protocol Example

- Transaction 1 accessing a book



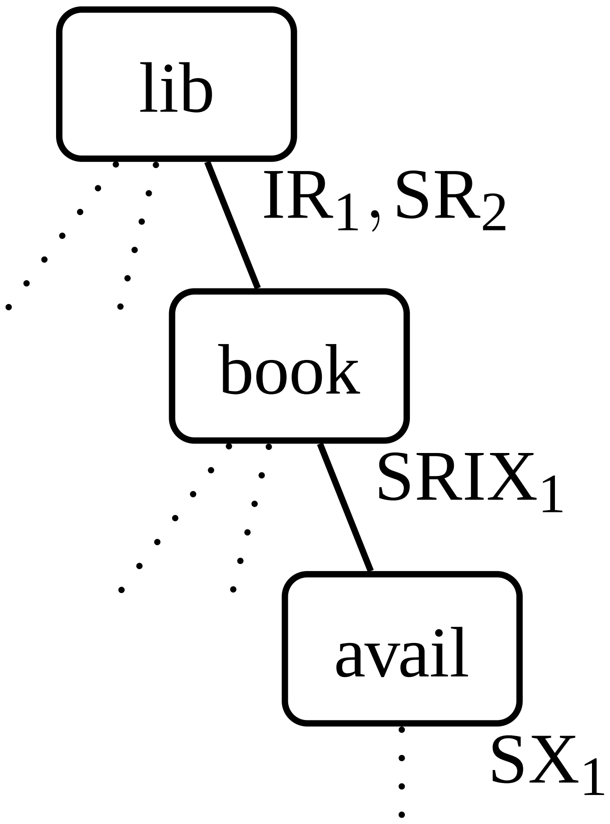
taDOM+ Protocol Example

- Transaction 1 accessing the book and Transaction 2 accessing all the books



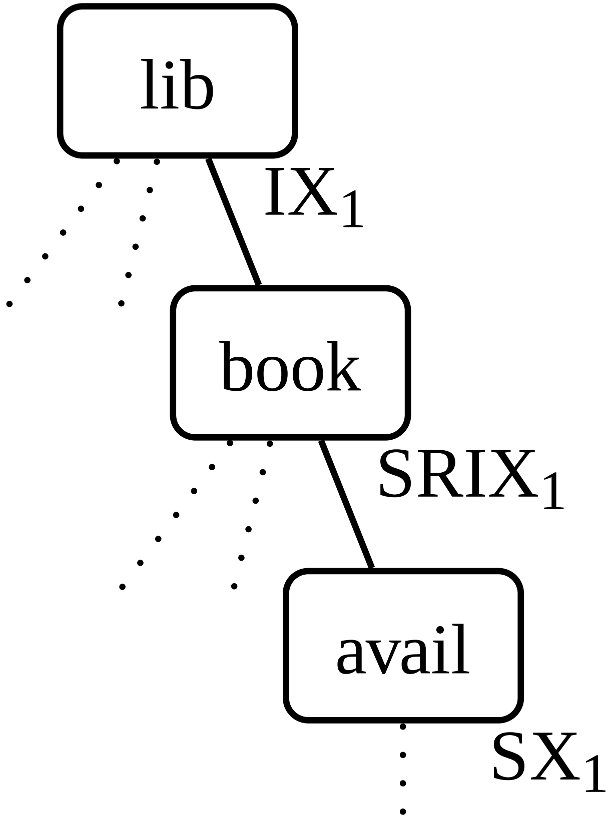
taDOM+ Protocol Example

- Transaction 2 accessing all the books and Transaction 1 trying to update availability information of the book



taDOM+ Protocol Example

- Transaction 1 updating the availability information of the book



Does the Protocol Work Correctly?

- Are the transactions isolated correctly?
- The existing taDOM protocols are extensively tested,
- but not proved correct.

Formal Verification

- proving or disproving the correctness of a system with respect to a certain property
- The behaviour of the system must be captured in a formal model.
- The correctness specification must be formalised.

Limits of Formal Verification

- Verifying systems with infinite state-space is generally undecidable.
- Verifying systems with large state-space is practically impossible.
- The most of the verification methods targeted to hardware systems made of components with bounded state-space.

Verification and taDOM+ Lock Managers

- The transactions and the nodes have an unbounded state-space.
- Only two verification methods can handle such systems:
  - data-independent induction (Creese, 2001),
  - induction theorem for ring protocols (Pyssysalo, 1996).
- The methods are not applicable for taDOM+ lock managers.

Parameterized Systems

- In practice, the state-space of a running application is bounded due to memory and other restrictions.
- typical approach: restrictions are modelled as parameters
- As parameters range over their domain, an infinite family of finite-state systems results.
- Finite-state verification tools can be used to check any finite subset of the family.

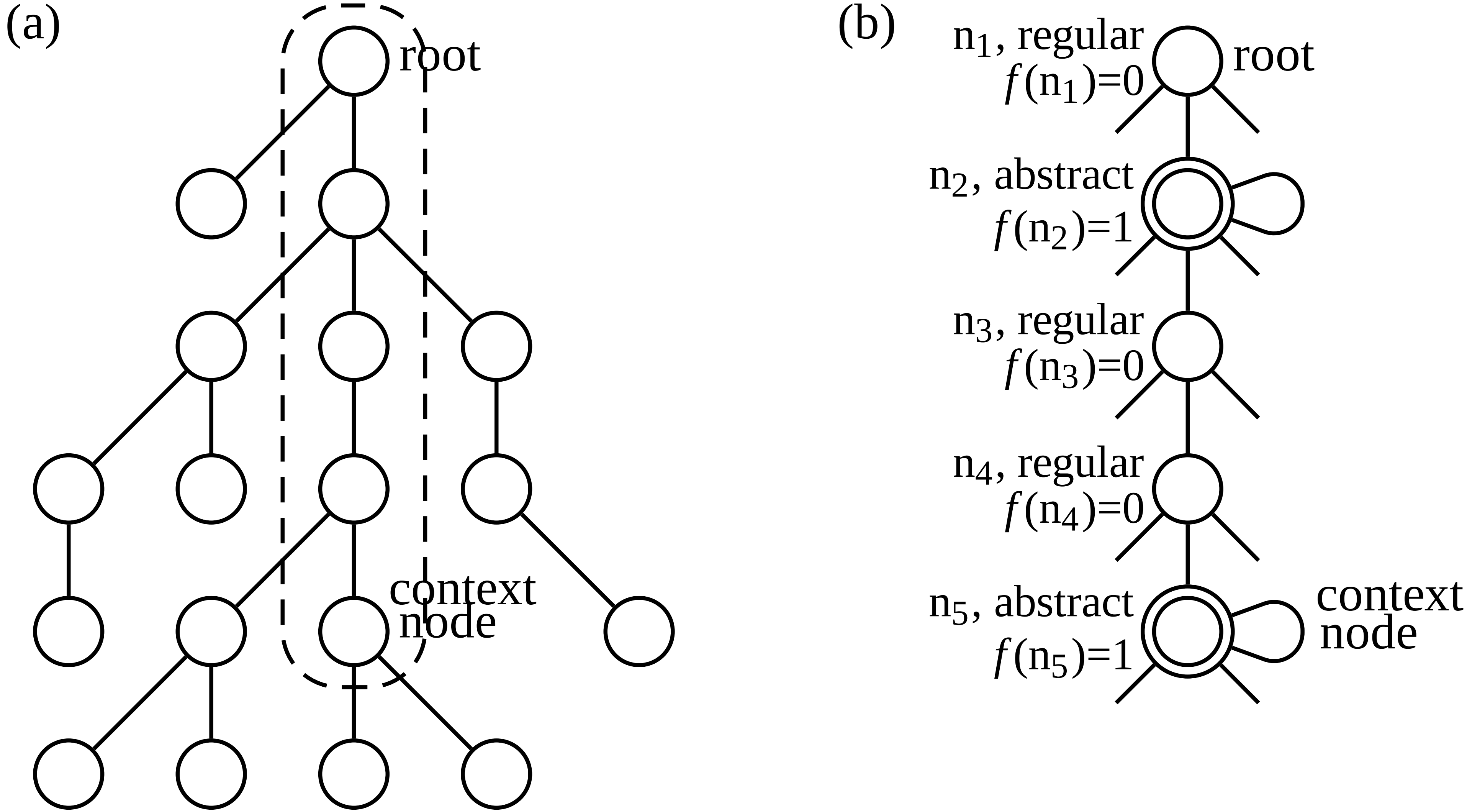
Verifying Infinite Families of Finite-State Systems

- Compactness results exist for
  - systems composed of similar fixed size processes (Attie, Emerson, 1998; Emerson, Kahlon, 2000),
  - rings of processes communicating through token passing (Emerson, Namjoshi, 1995; Emerson, Kahlon, 2004),
  - rings of Petri nets (Li, Suzuki, Yamashita, 1994; Lesens, Halbwachs, Raymond, 2001),
  - cache coherence protocols (Henzinger, Qadeer, Rayamani, 1999; Emerson, Kahlon 2003).

Modelling taDOM+ Lock Managers

- The number of the transactions is one parameter.
- Abstract nodes representing sequences of one or more successive regular nodes are introduced.
- An arbitrary node called the context is chosen.
- Database parameter describes the path from the root to the context node, other nodes are not explicitly modelled.

Abstracting Database



Properties

- We are interested in safety properties related to two arbitrary transactions and one arbitrary node called the context node.
  - safety property: absence of incorrect behaviour
- The property can refer to
  - the creation and the end of the transactions,
  - the beginning and the the end of the operations on the context node performed by the transactions.

Compactness Result

- Using only two transactions any two transactions of a bigger system with the same database parameter can be simulated.
- Using at most 2n regular and 2n+1 abstract nodes, where n is the number of different operations on nodes, the behaviour of a bigger system with two transactions can be simulated.
- The exact number of the nodes needed depends on the property.

Verifying taDOM2+ and taDOM3+

- The results are applied to the lock managers of taDOM2+ and taDOM3+ protocols and repeatable-read property.
- Repeatable-read property states that reading the same node within a transaction should give the same result unless the transaction itself has changed the contents of the node.

Verifying taDOM2+ and taDOM3+

- Repeatable-read property of the lock managers of taDOM2+ and taDOM3+ can be verified by checking all the instances with two transactions and at most 3 regular and 4 abstract nodes
- There are 18 instances per protocol to be checked.

Verifying taDOM2+ and taDOM3+

- taDOM2+ has 12 lock modes
  - The largest instance checked has 25 million states and 330 million transitions and took 30 minutes and 720MB of memory to complete.
- taDOM3+ has 20 lock modes
  - The largest instance checked has 120 million states and 1.4 billion transitions and took 180 minutes and 3.3GB of memory to complete.
- The lock managers of both the protocols were found to be true.

Model Limitations

- No data is modelled.
- Node inserts and removals are not modelled.
- The locks of a transaction are not released until the transaction ends.
- False negative answers are possible because of database abstraction.
  - Does not happen with real taDOM+ protocols.
  - False positives are not possible.

Topics of Future Research

- improving the model
  - allowing node inserts and removals
  - other operating modes, i.e. the locks could be released any time
- generalising the compactness results

Questions?

Thank You!