Verifying Programs on Relaxed Memory Models with focus on x86-TSO

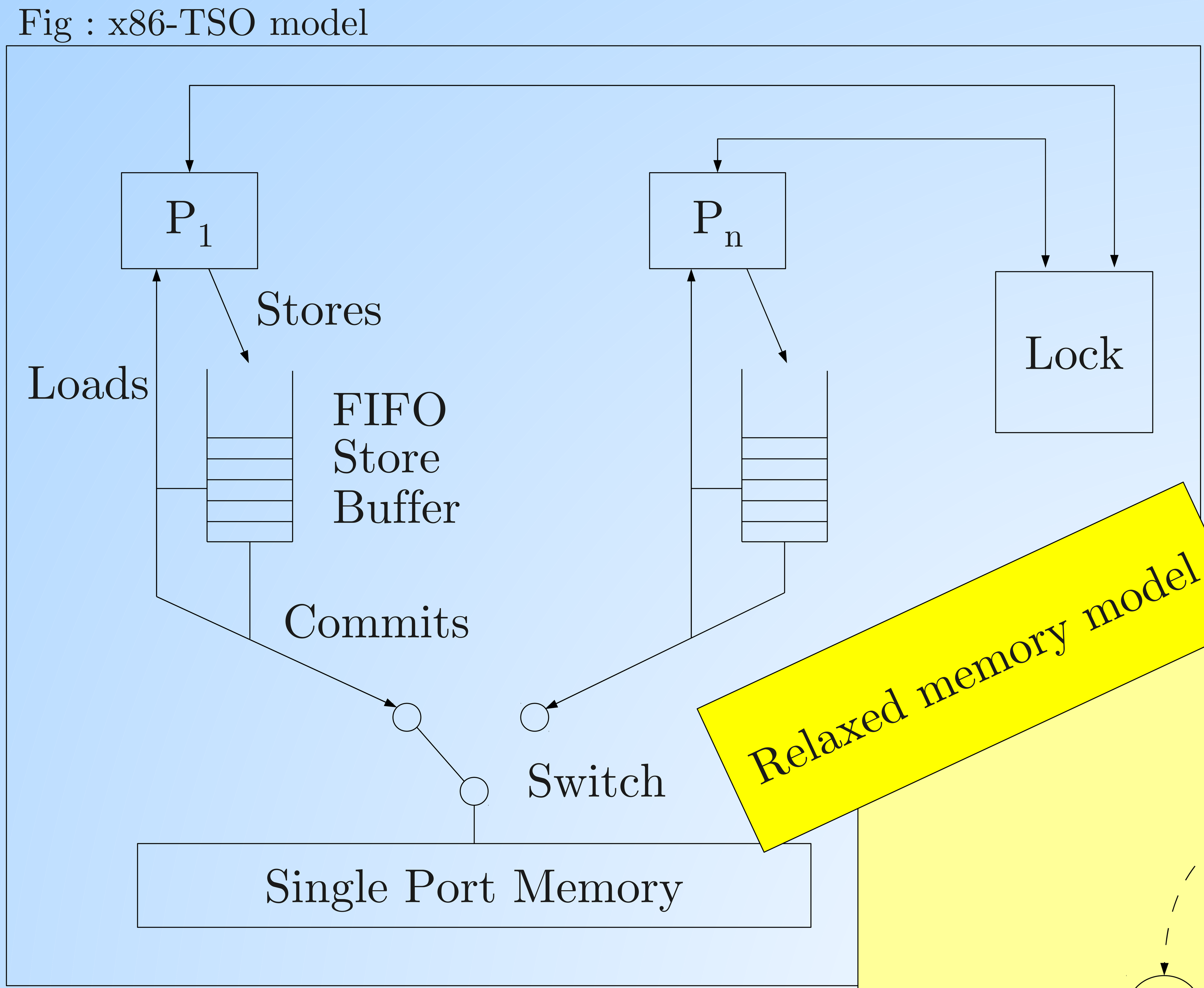
Alexander Linden

Institut Montefiore, Université de Liège, Belgium

The problem

Ensuring that concurrent programs remain correct when moved to multi-core processors implementing relaxed memory models (x86-TSO).

Fig : x86-TSO model



The approach : state space exploration and memory fence insertion

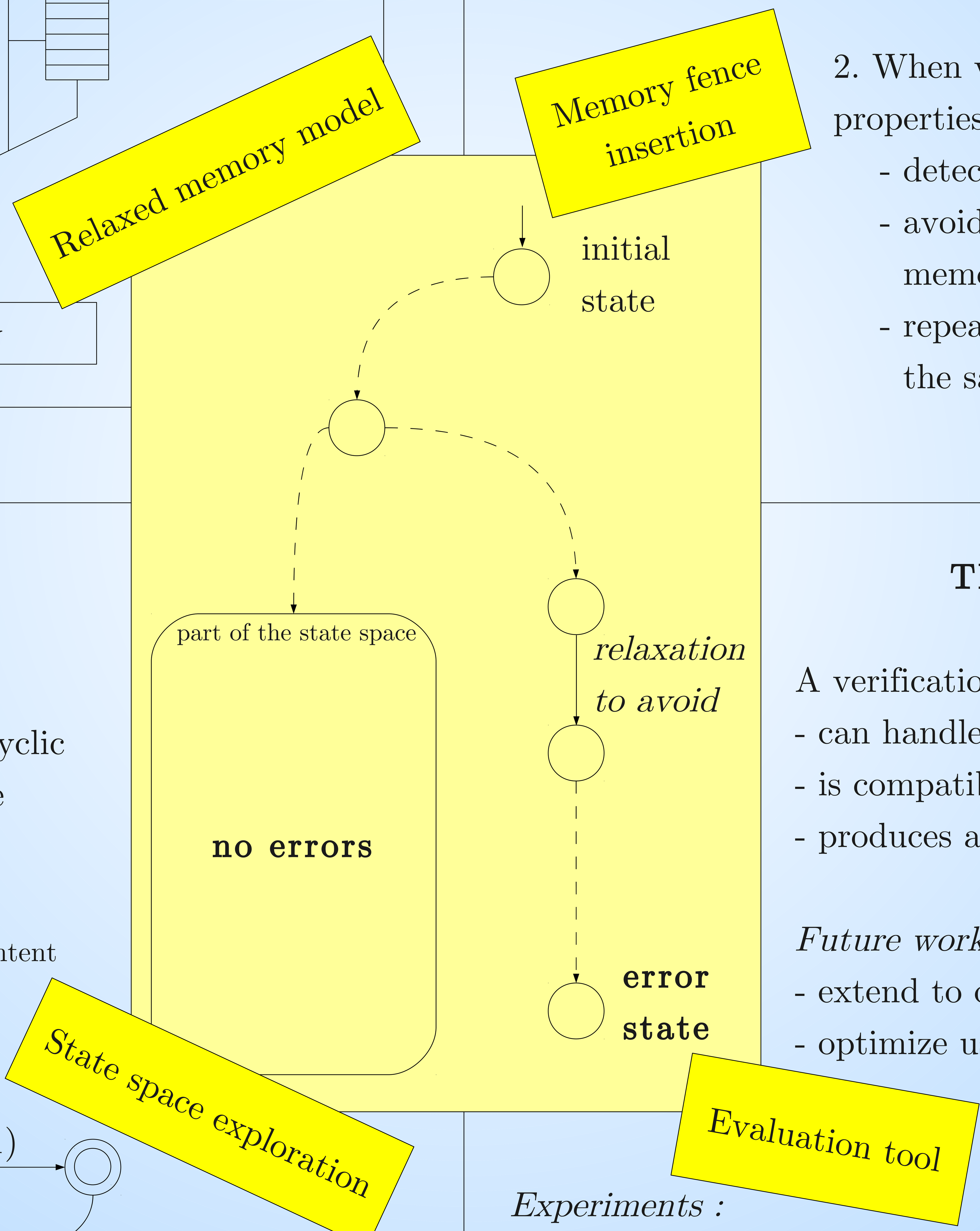
Start with a program that is correct (with respect to a safety property) under SC (the standard memory model).

Verify that the safety property still holds when the program is moved to a relaxed memory model and correct it as needed.

Procedure :

1. Explore the state space of the program, modelling the store buffers.

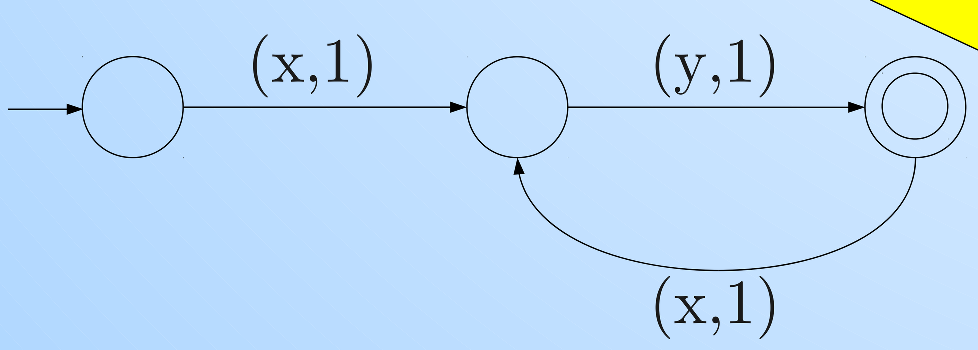
2. When violations of safety properties are found :
   - detect a problematic relaxation ;
   - avoid it by inserting a memory fence into the program ;
   - repeat this procedure until the safety property is satisfied.



The features of the approach

- allows the verification of cyclic programs by modelling the store buffers by automata.

Example : Unbounded buffer content (x,1)(y,1)(x,1)(y,1) ... (x,1)(y,1) is represented by the finite automaton



- limits the size of the state space by using partial-order reduction techniques (POR) :
  - persistent sets,
  - sleep sets.

The results

A verification tool that
- can handle cyclic programs,
- is compatible with POR,
- produces a correct program.

Future work :
- extend to other memory models,
- optimize use of POR.

Experiments :

Mutual Exclusion Algorithms			without err. correction		with error correction			
Program	entry	#P	#St	t(s)	#St	#iter	#f	t(s)
Dekker	single	2	118	0.84	92	3	2	0.80
Dekker	repeated	2	5468	12,7	213	5	4	0.41
Peterson	single	2	108	0.09	52	3	2	0.03
Peterson	repeated	2	400	0.58	54	3	2	0.05
Gen. Pet.	single	3	15476	44,4	1164	7	6	1.55
Bakery	single	2	775	0.58	340	5	4	0.15

linden@montefiore.ulg.ac.be