

Extending the Wait-free Hierarchy to Multi-Threaded Systems

Matthieu Perrin Achour Mostefaoui Grégoire Bonin

LS2N, Université de Nantes

PODC 2020

ACM Symposium on Principles of Distributed Computing

Virtual Conference

August 3-6, 2020

The Wait-free Hierarchy

Consensus number

- ▶ Consensus number $k \in \mathbb{N}$:
 - ▶ Wait-free universal for k processes
 - ▶ Not wait-free universal for $k + 1$ processes
- ▶ Consensus number ∞ :
 - ▶ Wait-free universal for k processes, for all k

Significance

- ▶ Objects with CN x cannot implement objects with CN $y > x$

The Wait-free Hierarchy in Multi-Threaded Systems

Multi-threaded systems

- ▶ Threads can be created dynamically
 - ▶ No bound on the number of threads in an execution
- ▶ Allocation of unbounded but **finite** arrays
 - ▶ How to allocate one shared register to each thread?

The iterator stack:

- ▶ Infinite consensus number
- ▶ Not universal in multi-threaded systems

Problem statement

How to compare the synchronization power of shared objects in multi-threaded systems?

The Wait-free Hierarchy in Multi-Threaded Systems

Multi-threaded systems

- ▶ Threads can be created dynamically
 - ▶ No bound on the number of threads in an execution
- ▶ Allocation of unbounded but **finite** arrays
 - ▶ How to allocate one shared register to each thread?

The iterator stack:

- ▶ Infinite consensus number
- ▶ Not universal in multi-threaded systems

Problem statement

How to compare the synchronization power of shared objects in multi-threaded systems?

The Wait-free Hierarchy in Multi-Threaded Systems

Multi-threaded systems

- ▶ Threads can be created dynamically
 - ▶ No bound on the number of threads in an execution
- ▶ Allocation of unbounded but **finite** arrays
 - ▶ How to allocate one shared register to each thread?

The iterator stack:

- ▶ Infinite consensus number
- ▶ Not universal in multi-threaded systems

Problem statement

How to compare the synchronization power of shared objects in multi-threaded systems?

Maximal number of processes in an execution:

M_1^n Classical model

- ▶ At most n processes (known to the developer)

M_1 Bounded arrival model

- ▶ The bound is known at initialization

M_2 Finite arrival model

- ▶ After some time, no new thread is started

M_3 Infinite arrival model

- ▶ New threads may keep arriving

Extended Wait-free Hierarchy

				Universal without infinite allocation?			
Arrival Models			Infinite				
			Finite				
			Bounded				
Universal with infinite allocation?	Infinite	Finite	Bounded	X	X	X	✓
	✓	✓	✓	X	X	✓	✓
	X	✓	✓	X	✓	✓	✓
	X	X	✓				
	X	X	X				
							Universal in multi-threaded systems

Filling the Hierarchy

				Universal without infinite allocation?			
Arrival Models			Infinite				
			Finite				
			Bounded				
Universal with infinite allocation?	Infinite	Finite	Bounded	X	X	X	✓
	✓	✓	✓	X	X	✓	✓
	X	✓	✓	X	✓	✓	✓
	X	X	✓				
	X	X	X				
				?	?	?	?
				?	?	?	?
				?	?	?	?
				?	?	?	?

Filling the Hierarchy

				Universal without infinite allocation?			
Arrival Models			Infinite				
			Finite				
			Bounded				
Universal with infinite allocation?	Infinite	Finite	Bounded	X	X	X	✓
	✓	✓	✓	X	X	✓	✓
	X	✓	✓	X	✓	✓	✓
	X	X	✓	?	?	?	?
	X	X	X	?	?	?	?

empty
(if universal without infinite allocation, still universal with infinite allocation)

Filling the Hierarchy

				Universal without infinite allocation?			
Arrival Models			Infinite				
			Finite				
			Bounded				
Universal with infinite allocation?	Infinite	Finite	Bounded	X	X	X	✓
				X	X	✓	✓
				X	✓	✓	✓
	✓	✓	✓	?	?	?	consensus
	X	✓	✓	?	?	?	empty (if universal without infinite allocation, still universal with infinite allocation)
	X	X	✓	?	?		
	X	X	X	?			

Filling the Hierarchy

				Universal without infinite allocation?			
Arrival Models			Infinite				
			Finite				
			Bounded				
Infinite	Finite	Bounded					
Universal with infinite allocation?	✓	✓	✓	?	?	?	consensus
	✗	✓	✓	?	?	iterator stack	
	✗	✗	✓	?	?		
	✗	✗	✗	?		empty (if universal without infinite allocation, still universal with infinite allocation)	

Filling the Hierarchy


				Universal without infinite allocation?			
Arrival Models		Infinite					
				Finite	Finite	Bounded	
		Infinite	Finite	Bounded			
Universal with infinite allocation?	✓	✓	✓		?	?	consensus
	✗	✓	✓	?	?	iterator stack	
	✗	✗	✓	finite consensus number	?	empty	
	✗	✗	✗			(if universal without infinite allocation, still universal with infinite allocation)	

Filling the Hierarchy

				Universal without infinite allocation?			
Arrival Models							
				Infinite	Finite	Bounded	
				Infinite	Finite	Bounded	
Universal with infinite allocation?	✓	✓	✓	✗	✗	✗	✓
	✗	✓	✓	✗	✗	✓	✓
	✗	✗	✓	✗	✓	✓	✓
	✗	✗	✗	✗	✓	✓	✓
	✓	✓	✓		?	?	consensus
	✗	✓	✓	empty	?	iterator stack	
	✗	✗	✓		?		
	✗	✗	✗			empty	
				Read/Write	T&S		(if universal without infinite allocation, still universal with infinite allocation)
							[Herlihy1991]

Filling the Hierarchy

				Universal without infinite allocation?			
Arrival Models							
Infinite				×	×	×	✓
Finite				×	×	✓	✓
Bounded				×	✓	✓	✓
Universal with infinite allocation?	✓	✓	✓		?	?	consensus
	×	✓	✓	empty	?	iterator stack	
	×	×	✓		?		
	×	×	×				
				Read/Write	(if universal without infinite allocation, still universal with infinite allocation)		


 [Herlihy1991]

Filling the Hierarchy

				Universal without infinite allocation?			
Arrival Models				Infinite	Finite	Infinite	Finite
				Infinite	Finite	Bounded	
Universal with infinite allocation?	✓	✓	✓	✗	✗	✗	✓
	✗	✓	✓	✗	✗	✓	✓
	✗	✗	✓	✗	✓	✓	✓
	✗	✗	✗	empty	?	?	consensus
	✗	✗	✗	empty	?	iterator stack	

(if universal without infinite allocation, still universal with infinite allocation)

[Herlihy1991]

Read/Write, T&S, ...

Filling the Hierarchy

				Universal without infinite allocation?			
Arrival Models		Infinite	\times	\times	\times	\checkmark	
		Finite	\times	\times	\checkmark	\checkmark	
Infinite	Finite	Bounded	\times	\checkmark	\checkmark	\checkmark	
Universal with infinite allocation?	\checkmark	\checkmark	\checkmark	binary consensus	binary consensus + iterator stack	consensus	
	\times	\checkmark	\checkmark		empty	window registers	iterator stack
	\times	\times	\checkmark	empty	empty	empty	
	\times	\times	\times	Read/Write	(if universal without infinite allocation, still universal with infinite allocation)		