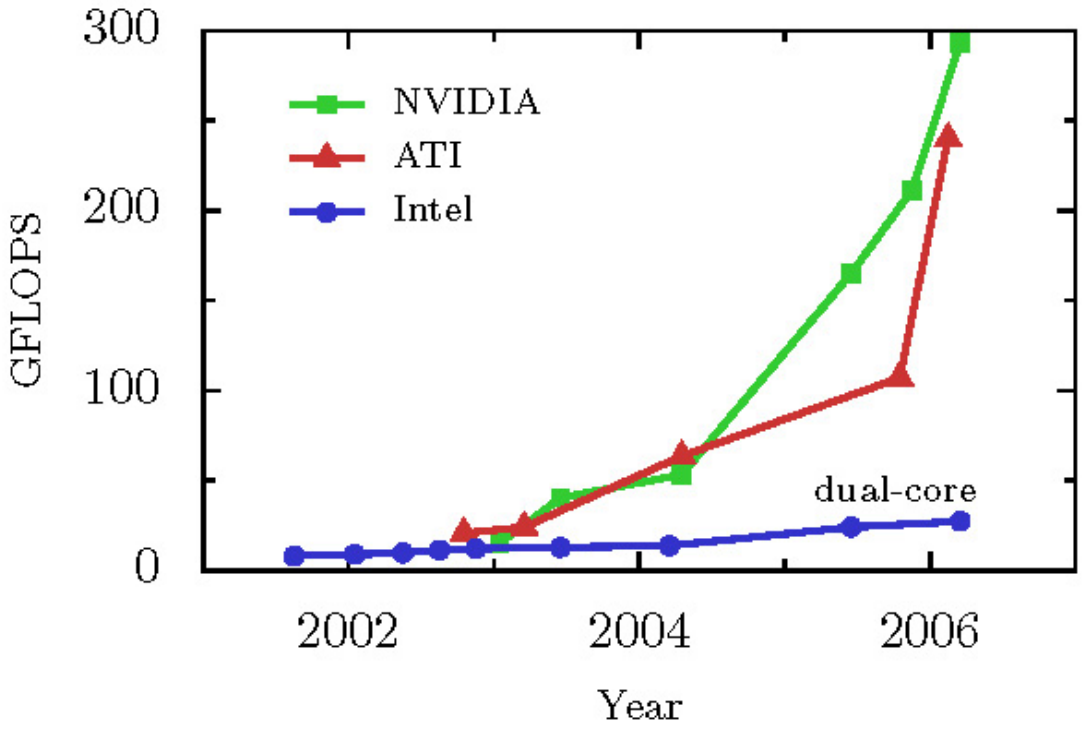
Wait-free Programming for General Purpose Computations on Graphics Processors

Phuong H. Ha (Univ. of Tromsø, Norway)
Philippas Tsigas (Chalmers Univ. of Tech., Sweden)
Otto J. Anshus (Univ. of Tromsø, Norway)

GPU: powerful and cheap



The chart is in *A Survey of General-Purpose Computation on Graphics Hardware* by J.D. Owens et al. [Computer Graphics Forum '07]

- GF 8800GTS 512
  - 624 GFLOPS
  - ~ $350
- Fastest supercomputer 1994 [top500.org]
  - 184 GFLOPS

⇒ GPGPU

- physics simulations,
- signal processing,
- geometric computing,
- databases
- etc.

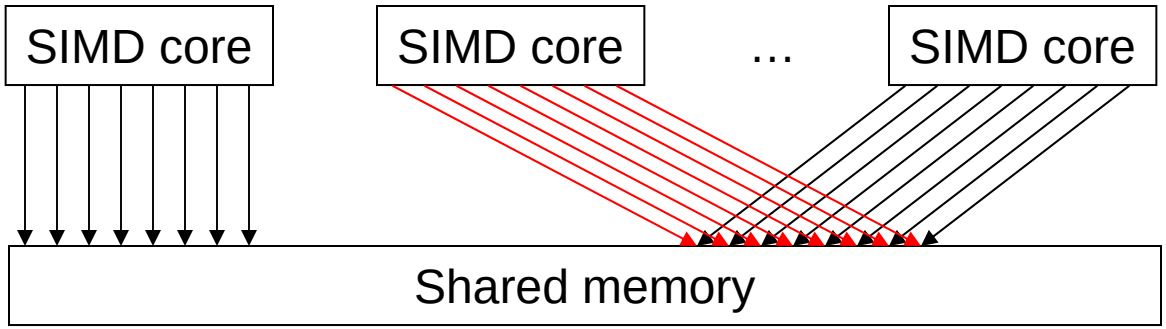
But …

- Lack of strong synchronization primitives
  - Pixels don't need to communicate

$\Rightarrow$ prevent GPU from being deployed more widely (e.g. conventional concurrent programming, lock-free/wait-free programming)

Can we construct strong synchronization mechanisms for GPU?

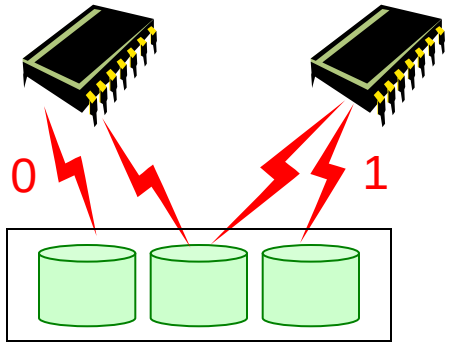
GPU Intrinsic features

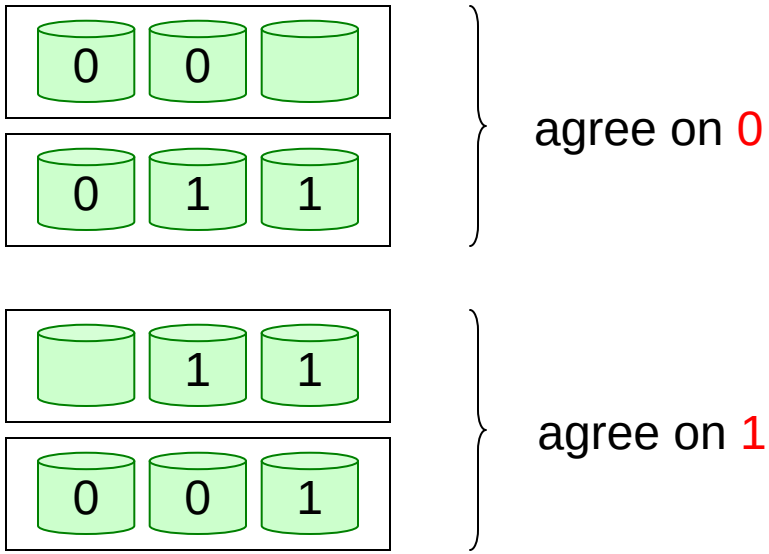


m-word assignment

Consensus using m-word assignment

- Wait-free (2m-2)-consensus [Herlihy, TOPLAS '91]
- Ex: 2-word assignment





short-lived consensus object !

Our main technical contribution

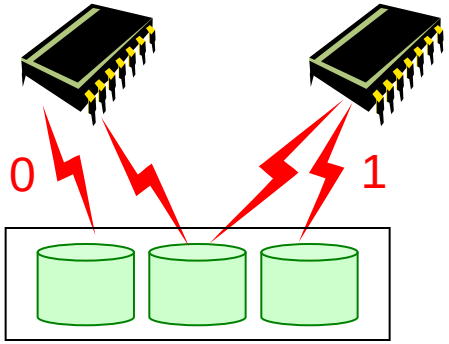
(2m-3)-resilient
long-lived
read-modify-write objects
using m-word assignment

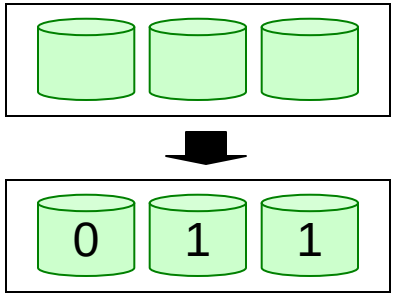
Road-map

- Wait-free long-lived consensus objects for (2m-2) processes.
- Wait-free long-lived read-modify-write (RMW) objects for (2m-2) processes
- (2m-3)-resilient long-lived RMW objects for any number of processes

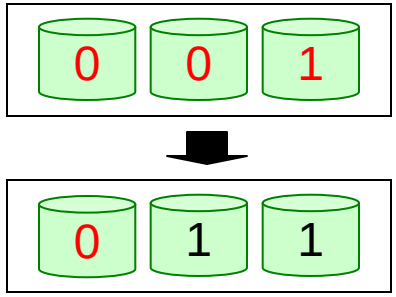
Wait-free long-lived (2m-2)-consensus (WF LLC)

WF LLC variables must be reusable





agree on 0



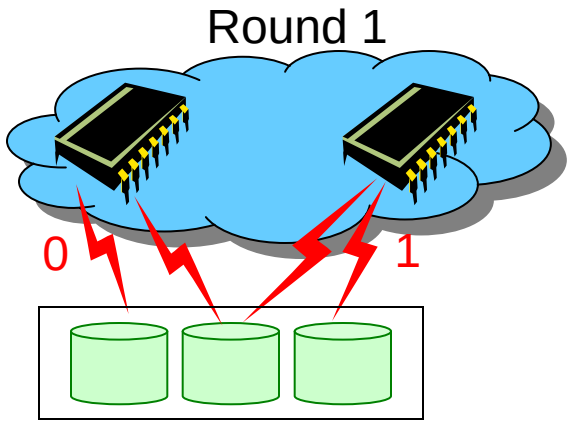
must agree on 1

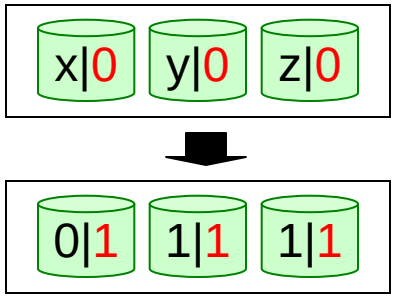
WF LLC: Key ideas

- Models:
  - each process p_i is associated with a round number
  - p_i gets a round number *r* only if round *(r-1)* has finished.

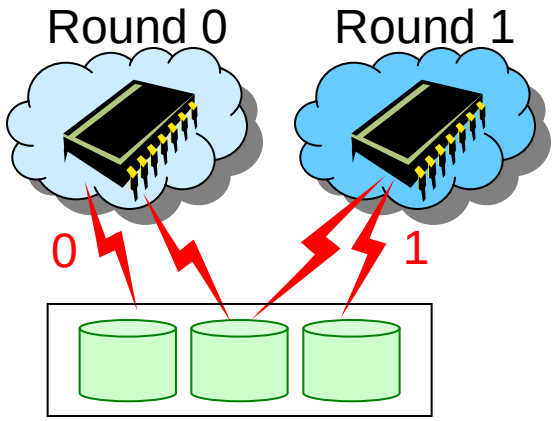
- WF LLC objects return
  - *nil* if p_i's round has finished
  - a value proposed in p_i's round.

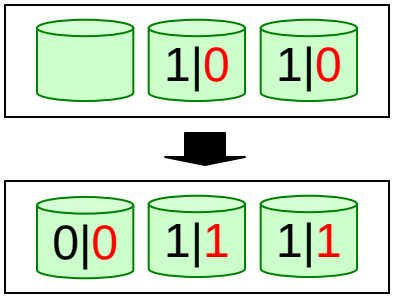
WF LLC: An illustration





agree on 0





- p_0 gets *nil*
- p_1 agrees on 1

WF LLC: Complexity

- Time complexity *O(N)*, where *N=(2m-2).*
  - vs. time complexity $O(N^2)$ of the WF *short-lived* consensus [TOPLAS '91].
  - Key idea: utilize the transitive property of the preceding order.
- Space complexity $O(N^2)$, the optimal.

Road-map

- Wait-free long-lived consensus objects for (2m-2) processes.
- Wait-free long-lived read-modify-write (RMW) objects for (2m-2) processes
- (2m-3)-resilient long-lived RMW objects for any number of processes

Wait-free RMW for $N=(2m-2)$ (WF RMW)

```
function RMW(X, f)

atomically {
    temp ← X;
    X ← f(X);
    return( temp);
}
```

WF RMW: Key idea 1

- Model:
  - Each process p_i executes one function f_i on X at a time
- Use WF LLC
  - p_i locally executes the functions of concurrent processes on a copy of X in a certain order.
  - p_i's proposal = responses to these functions.
  - Invoke WF LLC to achieve an agreement on their proposal.

But …

Such proposals are too big to be stored in one word

- How to achieve an agreement for (2m-2) processes as the consensus object does?

WF RMW: Key idea 2

- p_i's proposal = reference to the buffer containing responses

- Key property of the buffer:
  - response to f_k is kept unchanged until p_k submits a new function.

WF RMW: Complexity

- Time complexity *O(N)*,
  - Each process invokes WF LLC at most 2 times to get a response
- Space complexity $O(N^2)$, the optimal.

Road-map

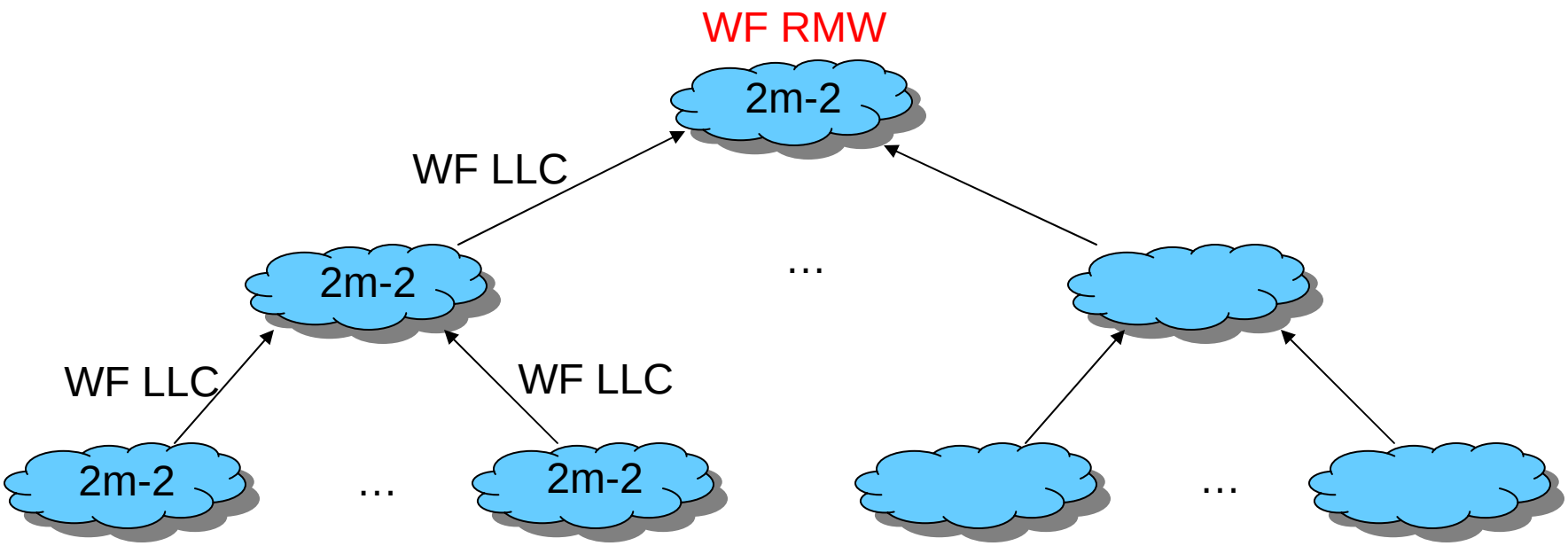
- Wait-free long-lived consensus objects for (2m-2) processes.
- Wait-free long-lived read-modify-write (RMW) objects for (2m-2) processes
- (2m-3)-resilient long-lived RMW objects for any number of processes

(2m-3)-resilient RMW

What if N > 2m-2?

WF LLC and WF RMW no longer work!

(2m-3)-resilient RMW: Key idea



- Balance tree with degree *(2m-2)*
- (2m-3)-resilient, the optimal [Chandra et al., SIAM J. Comput. '04]

Conclusions

- Design a set of universal synchronization objects for lock-free/wait-free programming on GPU
  - Wait-free long-lived (2m-2)-consensus using m-word assignment.
    - Time complexity: *O(N)* vs. $O(N^2)$ of the short-lived in [TOPLAS '91]
    - Space complexity: $O(N^2)$, the optimal.
  - Wait-free long-lived read-modify-write (RMW) objects for (2m-2) processes.
    - Time complexity *O(N)*.
    - Space complexity $O(N^2)$, the optimal.
  - (2m-3)-resilient long-lived RMW objects for any number of processes

Thank you for your attention!