Cellular automata (CA) halo exchange and Fortran coarrays

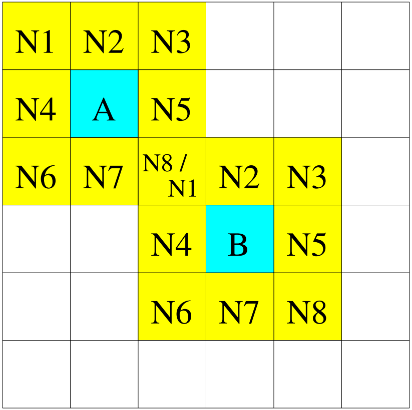
A. Shterenlikht
Department of Mechanical Engineering
The University of Bristol, Bristol BS8 1TR, UK
mexas@bris.ac.uk

Hartree 2016 Summer School on Engineering Simulation,
30-JUN-2016

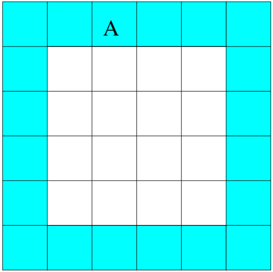
CA concurrency (parallelism)

- New cell states, $(i+1)$, depend only on old cell states, (i).
- State of cell A, S_A: $S_A(i+1) = f(S_A(i), S_1(i) \dots S_8(i))$.
- Cells A and B can be updated **concurrently** or **in parallel**.
- Need access only to this cell and its neighbourhood cells.

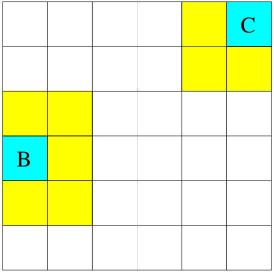
Example: 2D, Moore's neighbourhood, 8 neighbours (yellow) for any cell (cyan).



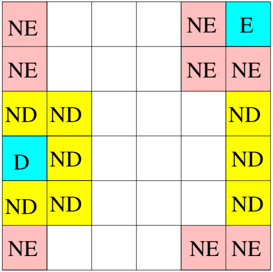
Boundary cells - many possibilities



1. Fixed.
Do not change state, $S_A(i+1) = S_A(i)$.



2. Special neighbourhoods.
E.g. edge cells B or corner cells C.
Rarely used because not all cells are equal.

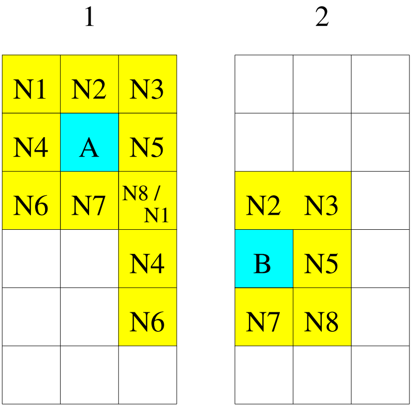


3. Self-similar (periodic or wrap-around) boundaries. Top continues as bottom, left continues as right, etc.

CA self-similar boundaries

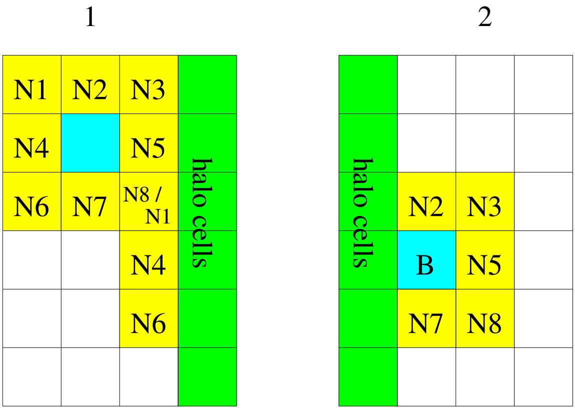
NE				NE	E	NE				NE	E	NE				NE	E
NE				NE	NE	NE				NE	NE	NE				NE	NE
ND	ND				ND	ND	ND				ND	ND	ND				ND
D	ND				ND	D	ND				ND	D	ND				ND
ND	ND				ND	ND	ND				ND	ND	ND				ND
NE				NE	NE	NE				NE	NE	NE				NE	NE
NE				NE	E	NE				NE	E	NE				NE	E
NE				NE	NE	NE				NE	NE	NE				NE	NE
ND	ND				ND	ND	ND				ND	ND	ND				ND
D	ND				ND	D	ND				ND	D	ND				ND
ND	ND				ND	ND	ND				ND	ND	ND				ND
NE				NE	NE	NE				NE	NE	NE				NE	NE
NE				NE	E	NE				NE	E	NE				NE	E
NE				NE	NE	NE				NE	NE	NE				NE	NE
ND	ND				ND	ND	ND				ND	ND	ND				ND
D	ND				ND	D	ND				ND	D	ND				ND
ND	ND				ND	ND	ND				ND	ND	ND				ND
NE				NE	NE	NE				NE	NE	NE				NE	NE

CA calculations in parallel



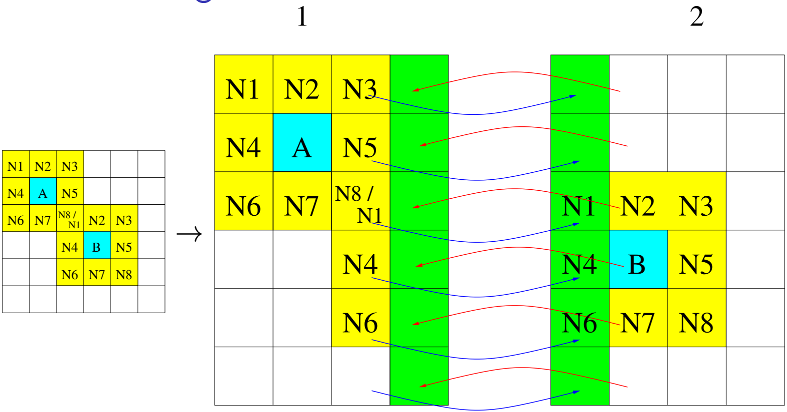
- Example: 2 processes (e.g. threads, MPI processes, etc.)
- Cell *A* is updated on process 1.
- Cell *B* cannot be calculated by process 2, because some neighbourhood cells are stored on process 1.
- Solution - halo cells.

CA halos



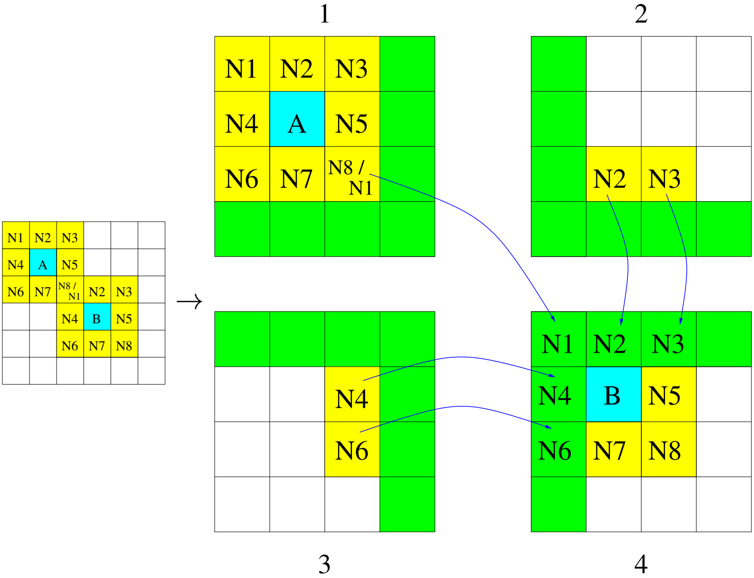
- Halo (ghost) cells are added beyond any new boundary.
- Halo cells are used to create a neighbourhood for all new boundary cells.

CA halo exchange 1D



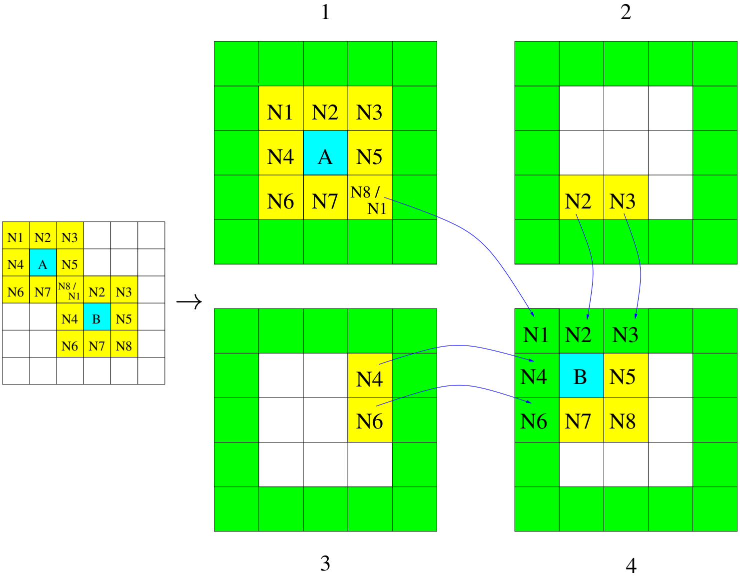
- Boundary CA cells from one process are copied into the halo cells on the matching process.
- Blue arrows copy boundary CA cells from process 1 into the halo cells on process 2.
- Red arrows copy boundary CA cells from process 2 into the halo cells on process 1.

CA halo exchange 2D



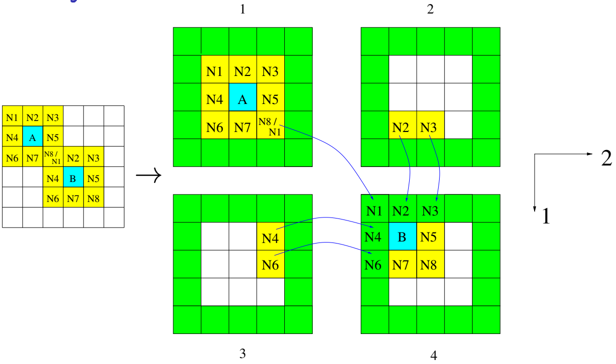
Only the necessary halo transfer operations are shown.

CA halo exchange + self-similar boundary 2D



The same solution is used for halo cells and for self-similar boundaries.

CA arrays in Fortran



- CA array is 6×6:

```
integer :: ca( 6 , 6 )
```

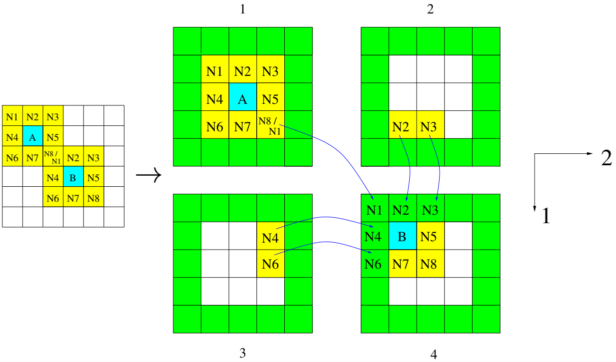
- CA array on each process is 3×3:

```
integer :: ca( 3 , 3 )
```

- CA + halo array is 5×5 on each process:

```
integer :: ca( 5 , 5 )
```

CA arrays in parallel: **Fortran 2008 coarrays**



to coarrays course

```
integer, allocatable :: ca(:,:)[:,:] ! coarray
integer :: pos(2)
allocate( ca(0:4,0:4) [2,*] ) ! +2 halo cells
pos = this_image( ca )              ! img grid pos.
if ( pos(1) .ne. 1 ) &              ! halo exchange
  ca(0,1:3) = ca(3,1:3) [ pos(1)-1, pos(2) ]
```