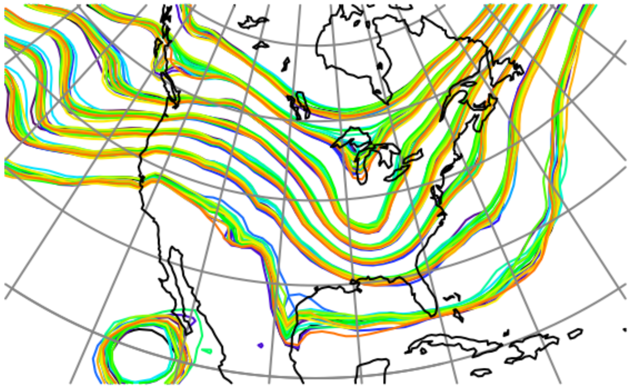
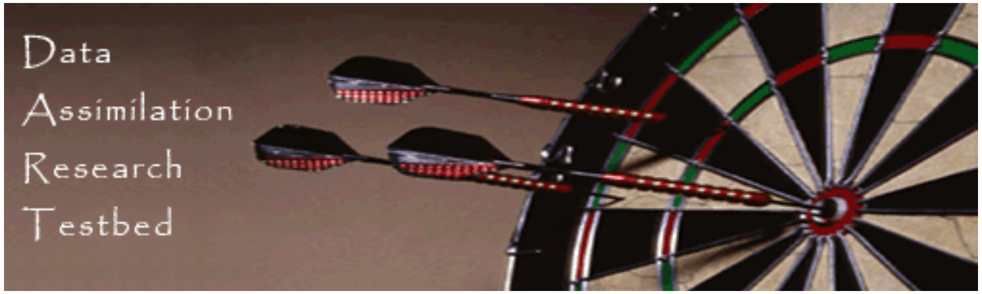


Parallelization Challenges for Ensemble Data Assimilation



Helen Kershaw

Institute for Mathematics Applied to Geophysics,
National Center for Atmospheric Research
Email: hkershaw@ucar.edu

What am I going to talk about?

- What’s ensemble data assimilation?
- What’s DART?
- What’s parallel about DART?
- What’s not so parallel about DART?
  - Data decomposition
  - IO
  - Algorithm and communication
- Software engineering concerns

What’s ensemble data assimilation?

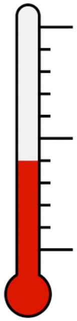
Ensemble Data Assimilation

group of model forecasts

Ensemble Data Assimilation

group of model forecasts

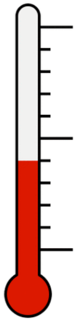
Measurements

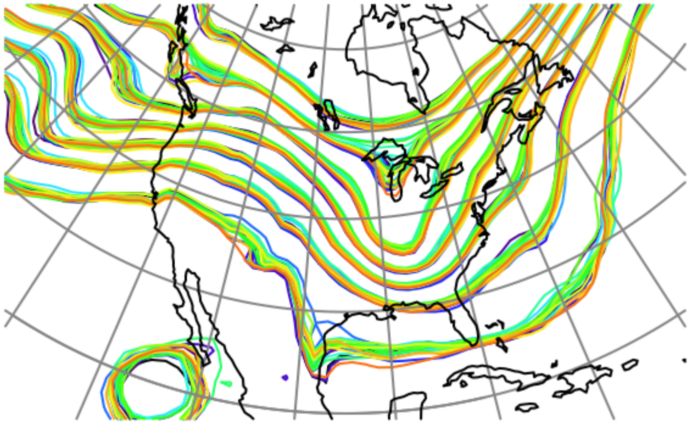


Ensemble Data Assimilation

group of model forecasts

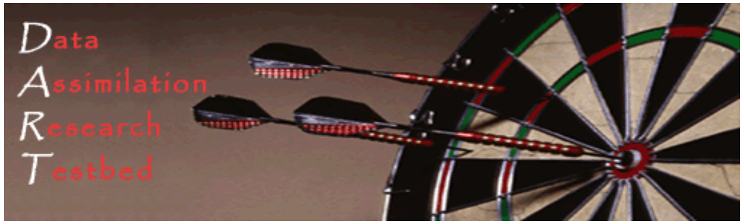
Measurements





Improved estimate

What’s DART?



DART is used at:

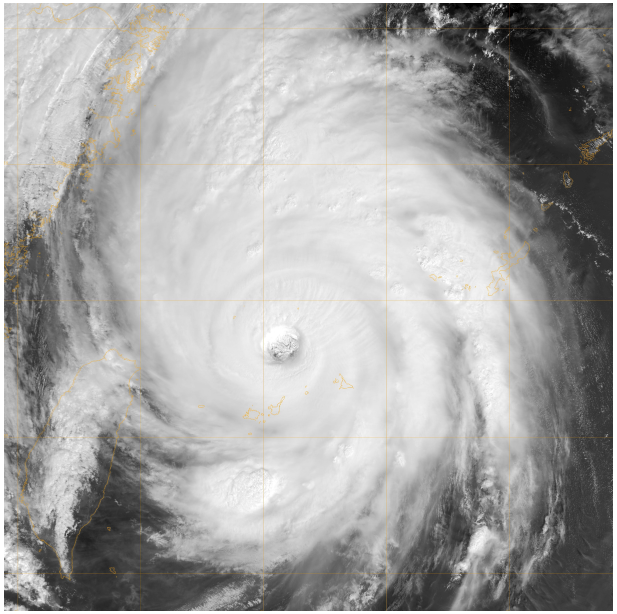
43 UCAR member universities

More than 100 other sites

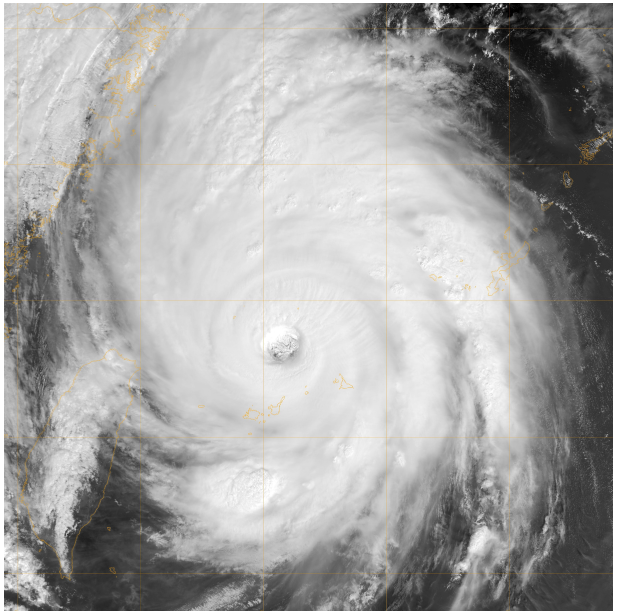
- Public domain software for Data Assimilation
  - Well-tested, portable, extensible, free!
- Models
  - Toy to HUGE
- Observations
  - Real, synthetic, novel
- An extensive Tutorial
  - With examples, exercises, explanations
- People: The DAReS Team

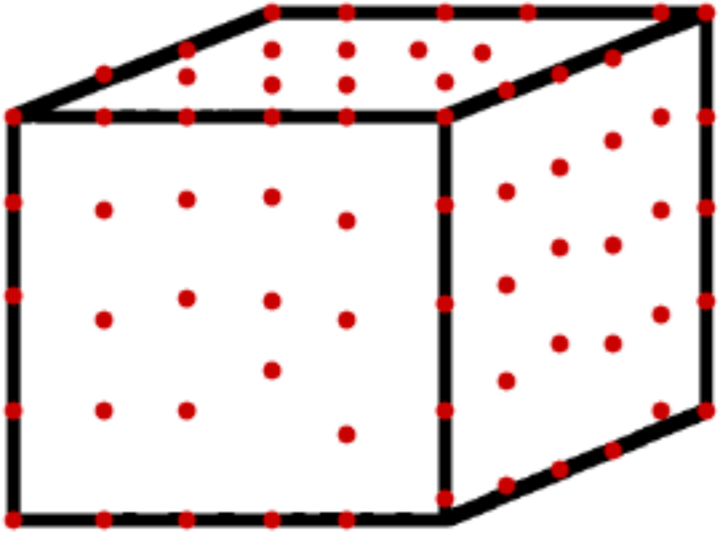
NSSL
UNIVERSITY OF WASHINGTON
CIRES
THE UNIVERSITY OF UTAH
W
VT
中央氣象局 Central Weather Bureau
GFDL Geophysical Fluid Dynamics Laboratory Princeton, NJ
NOAA NATIONAL OCEANIC AND ATMOSPHERIC ADMINISTRATION U.S. DEPARTMENT OF COMMERCE
PENNSTATE 1855
CALIFORNIA INSTITUTE OF TECHNOLOGY 1891
JPL Jet Propulsion Laboratory California Institute of Technology
NCAR NATIONAL CENTER FOR ATMOSPHERIC RESEARCH
NSF

The State

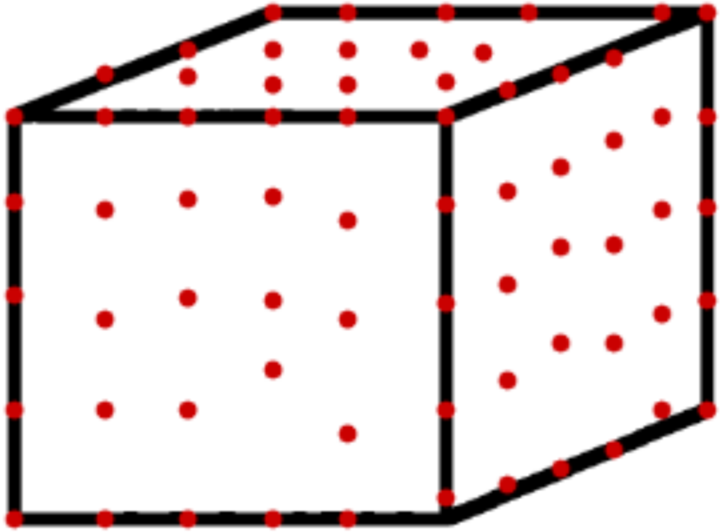


The State

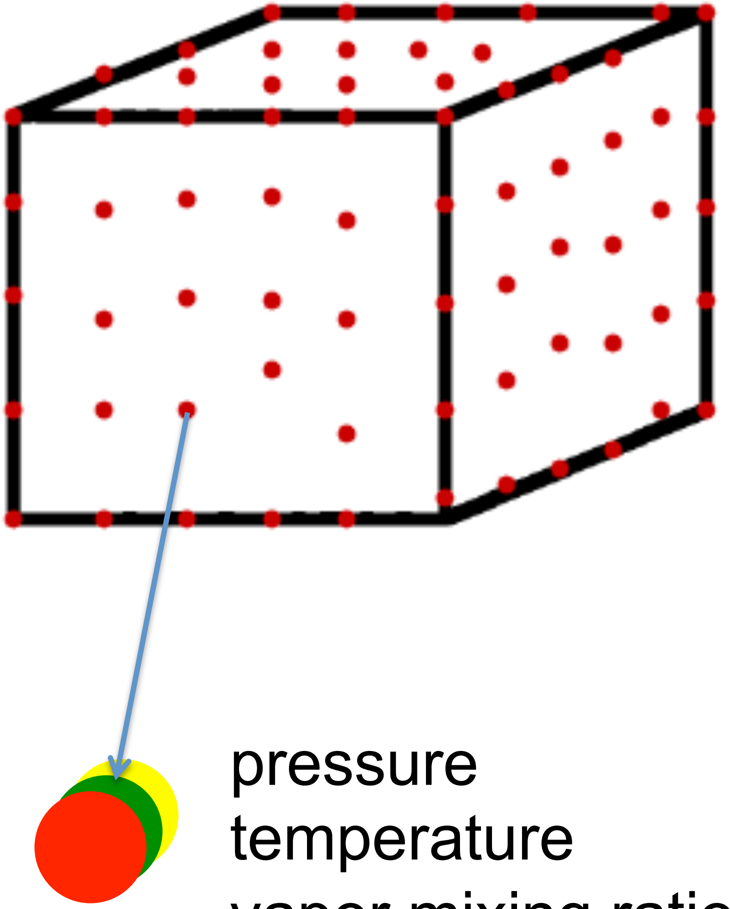




The State



The State

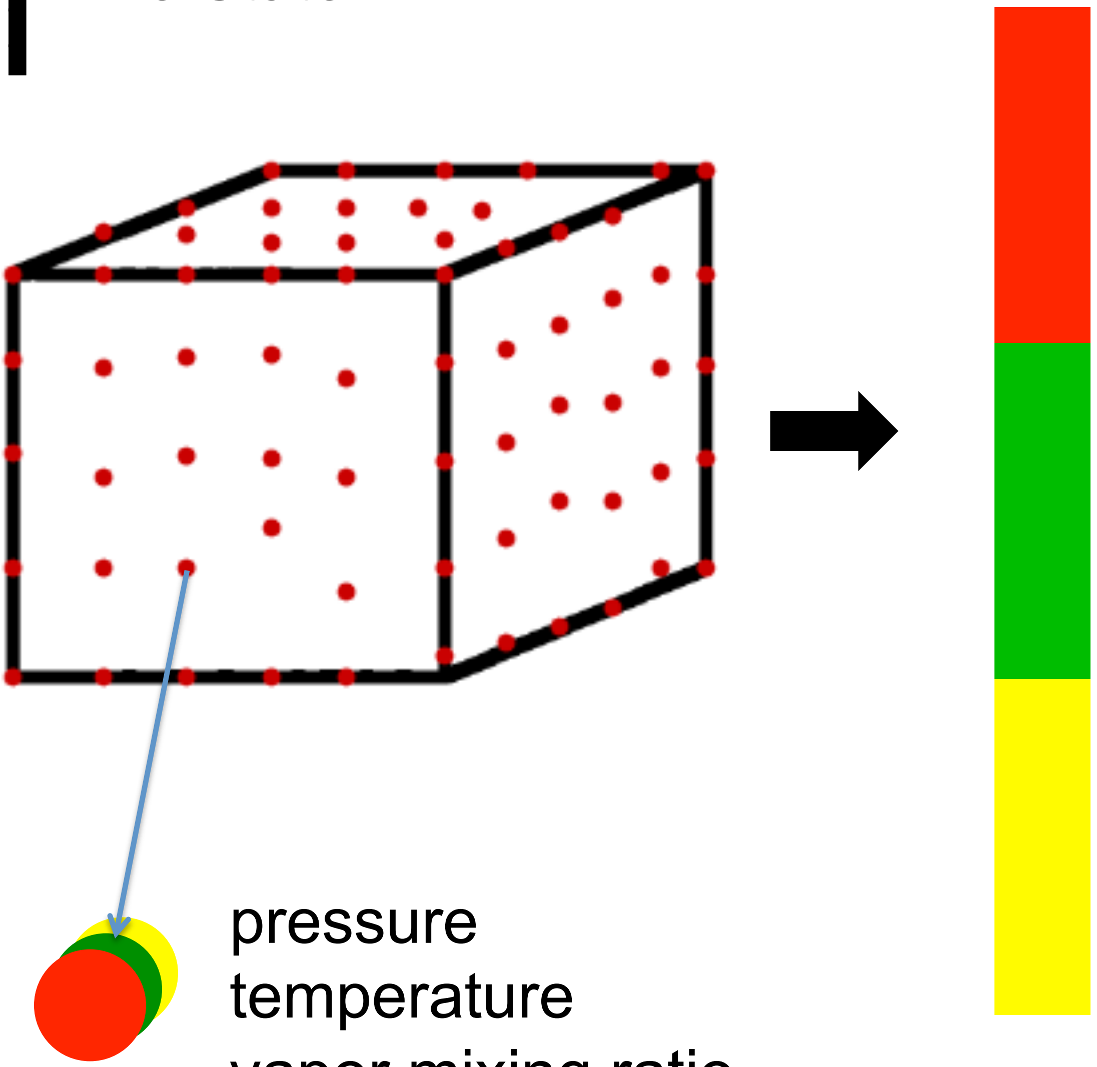


pressure

temperature

vapor mixing ratio

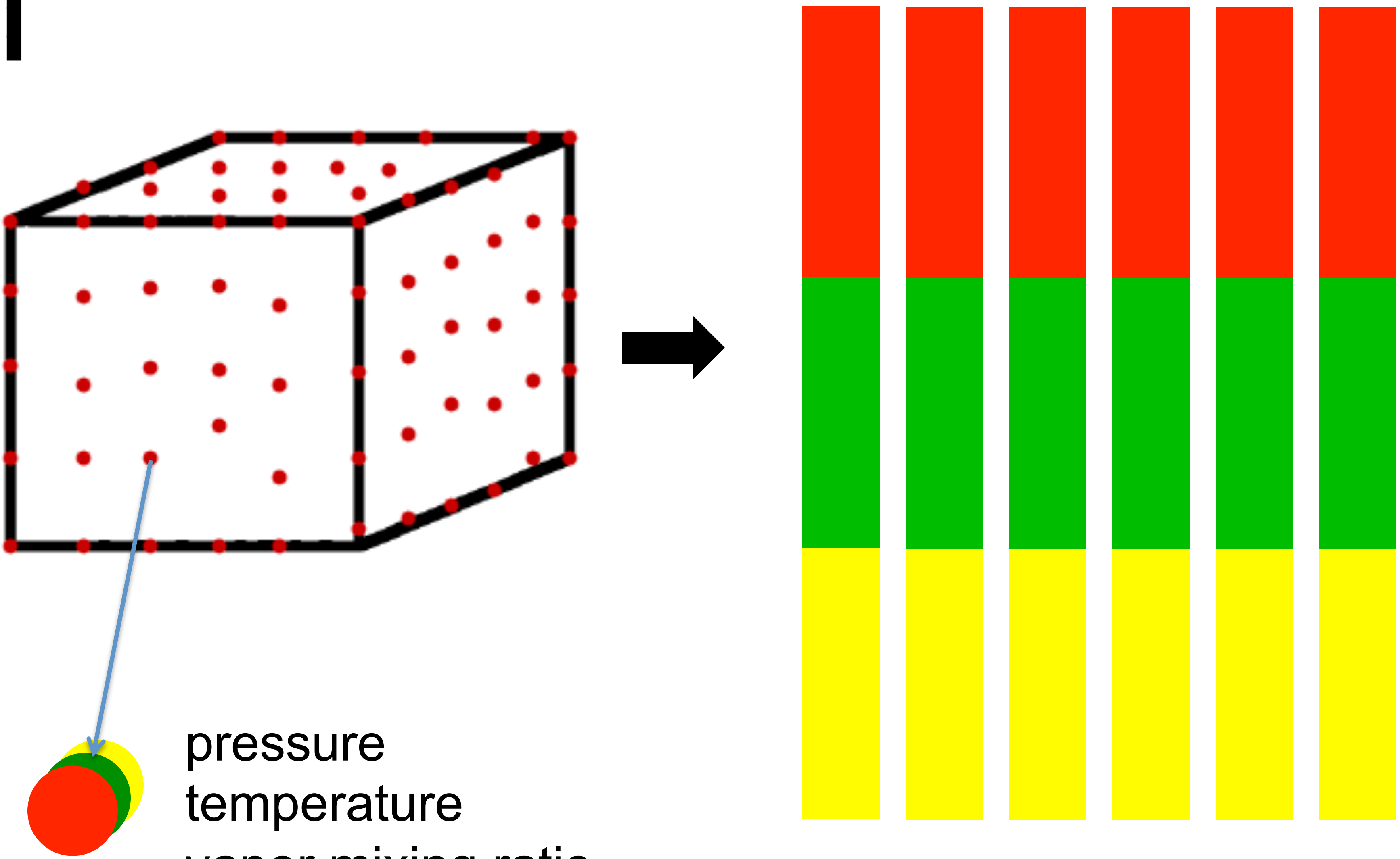
The State



pressure
temperature
vapor mixing ratio

DART state vector

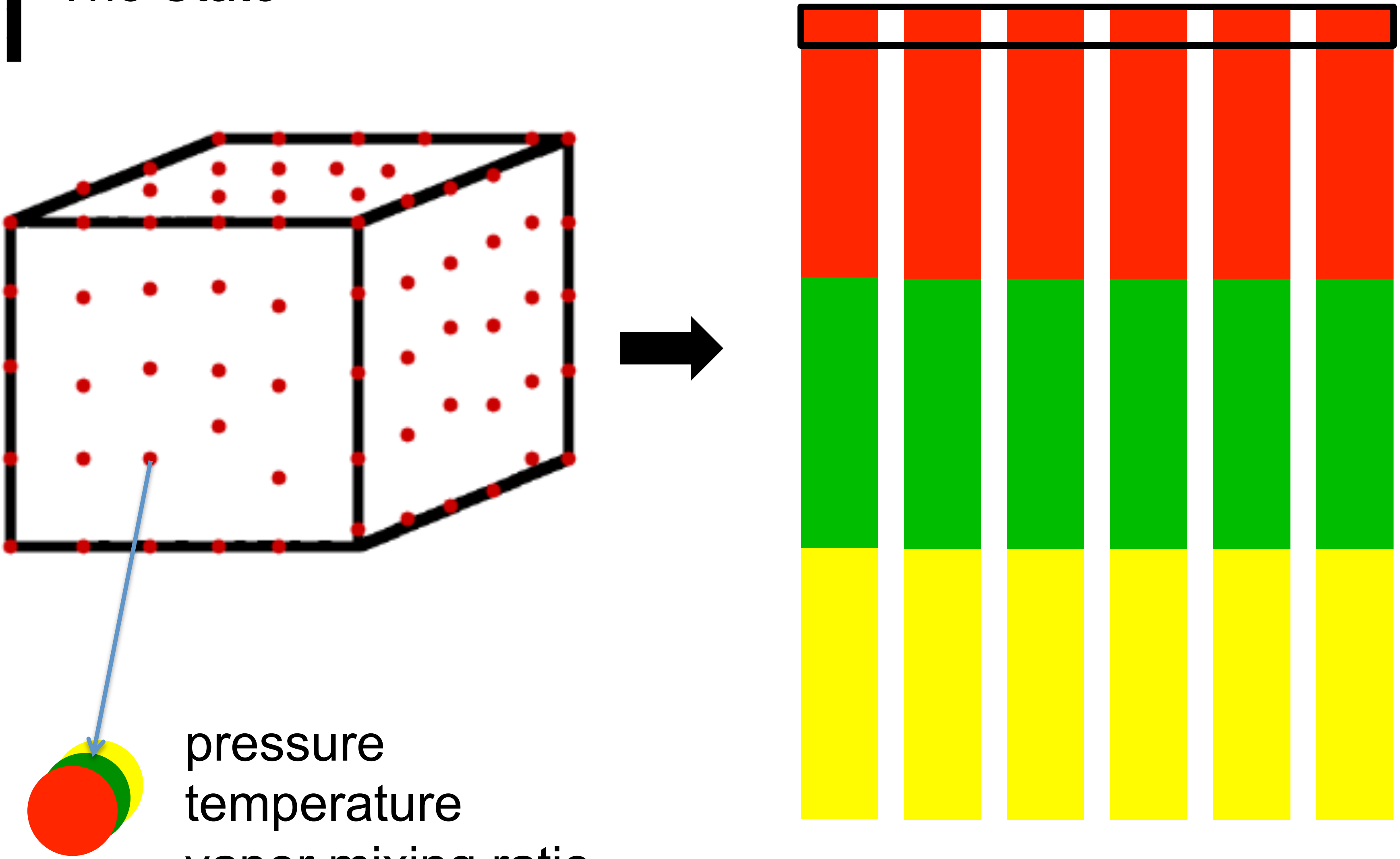
The State



pressure
temperature
vapor mixing ratio

multiple copies

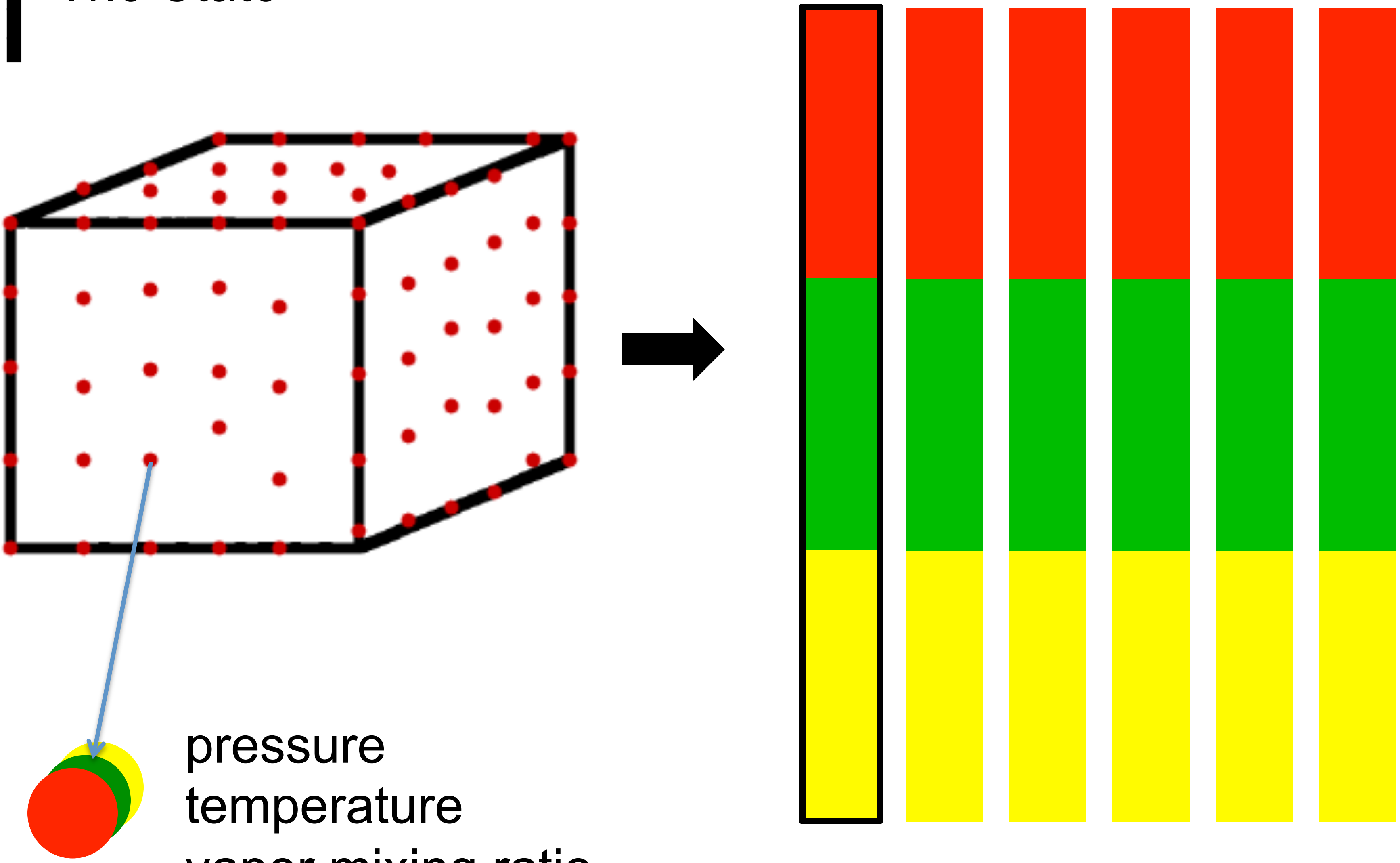
The State



pressure
temperature
vapor mixing ratio

multiple copies

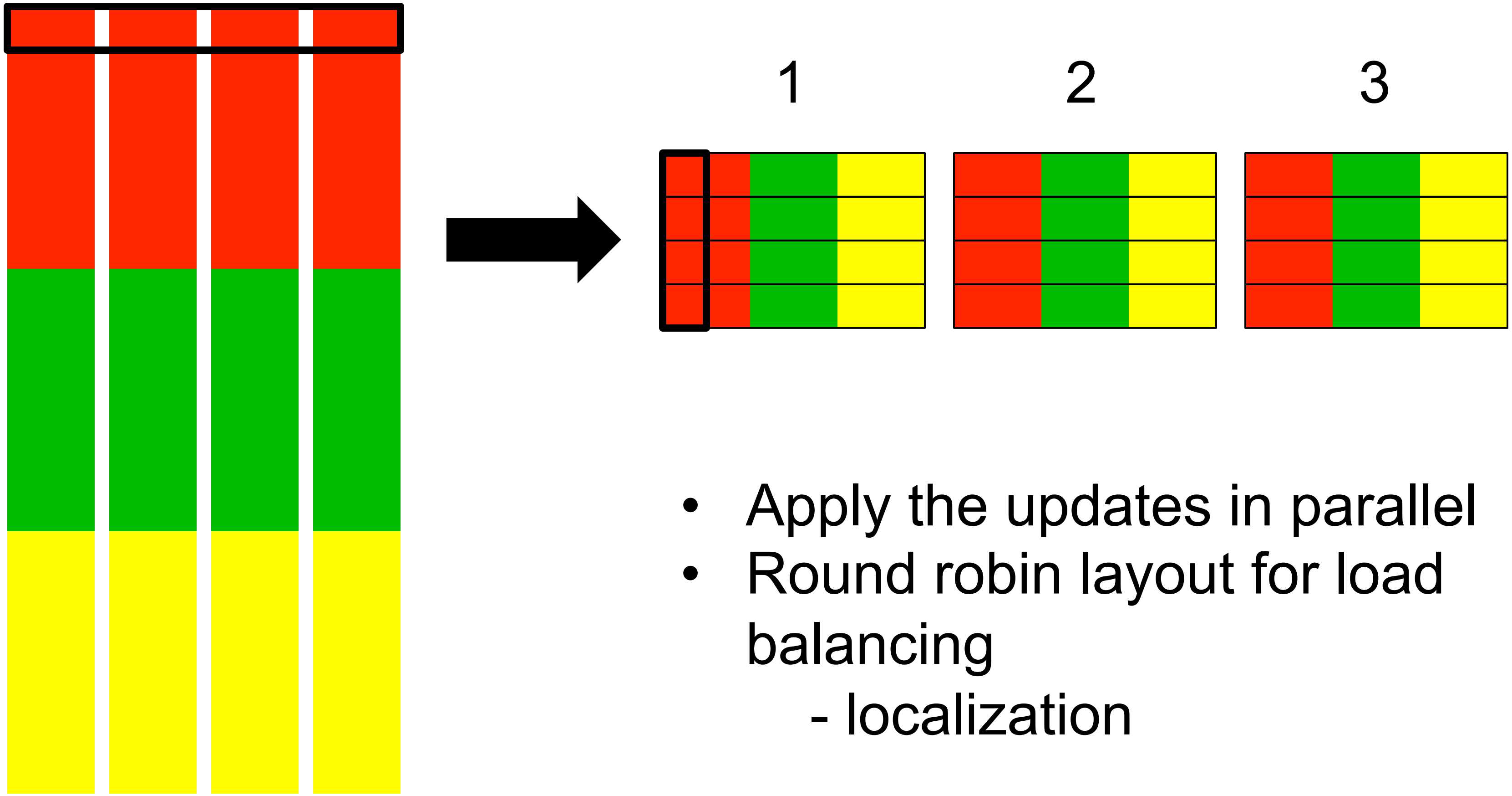
The State



pressure
temperature
vapor mixing ratio

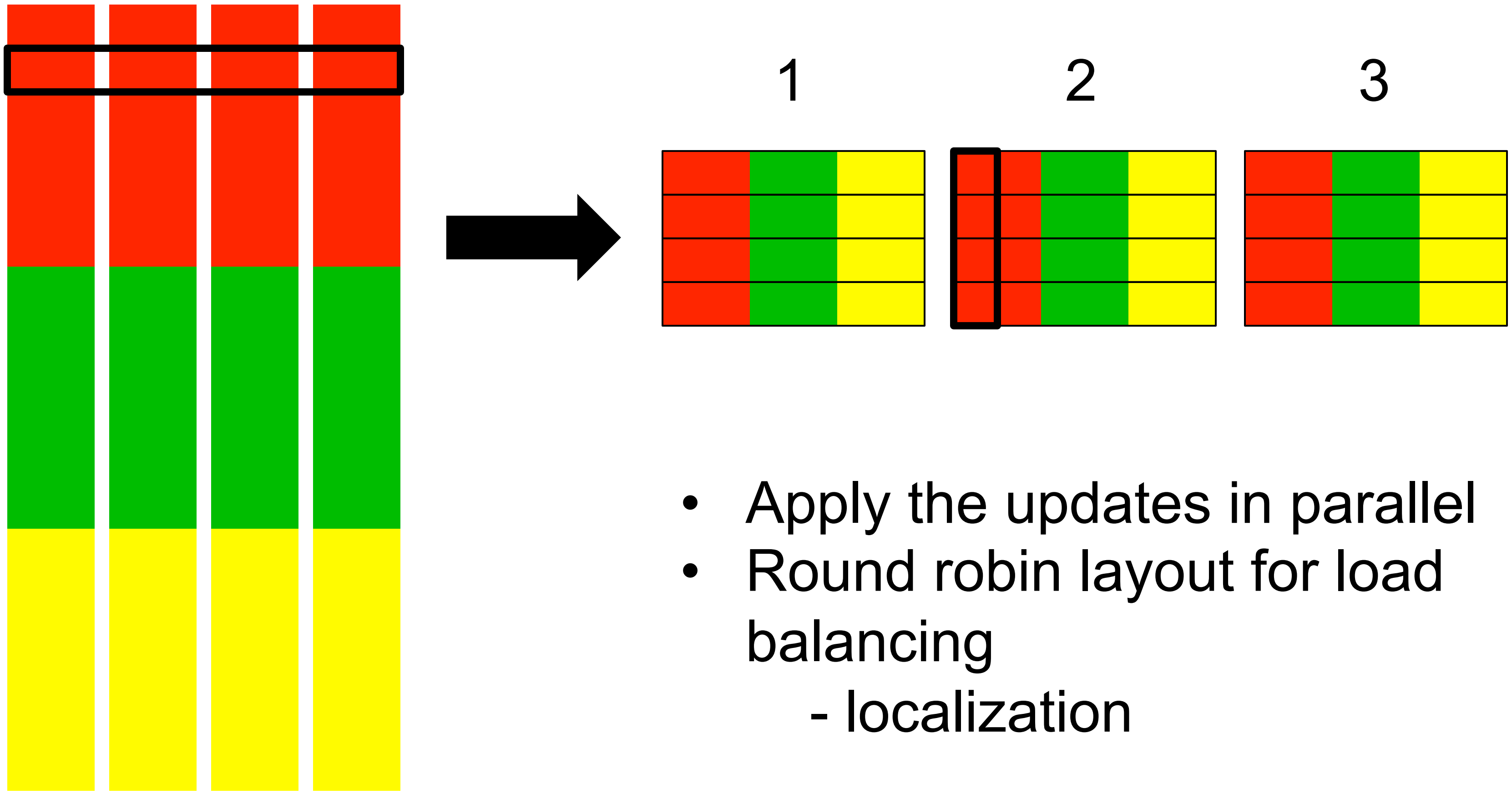
multiple copies

Assimilation



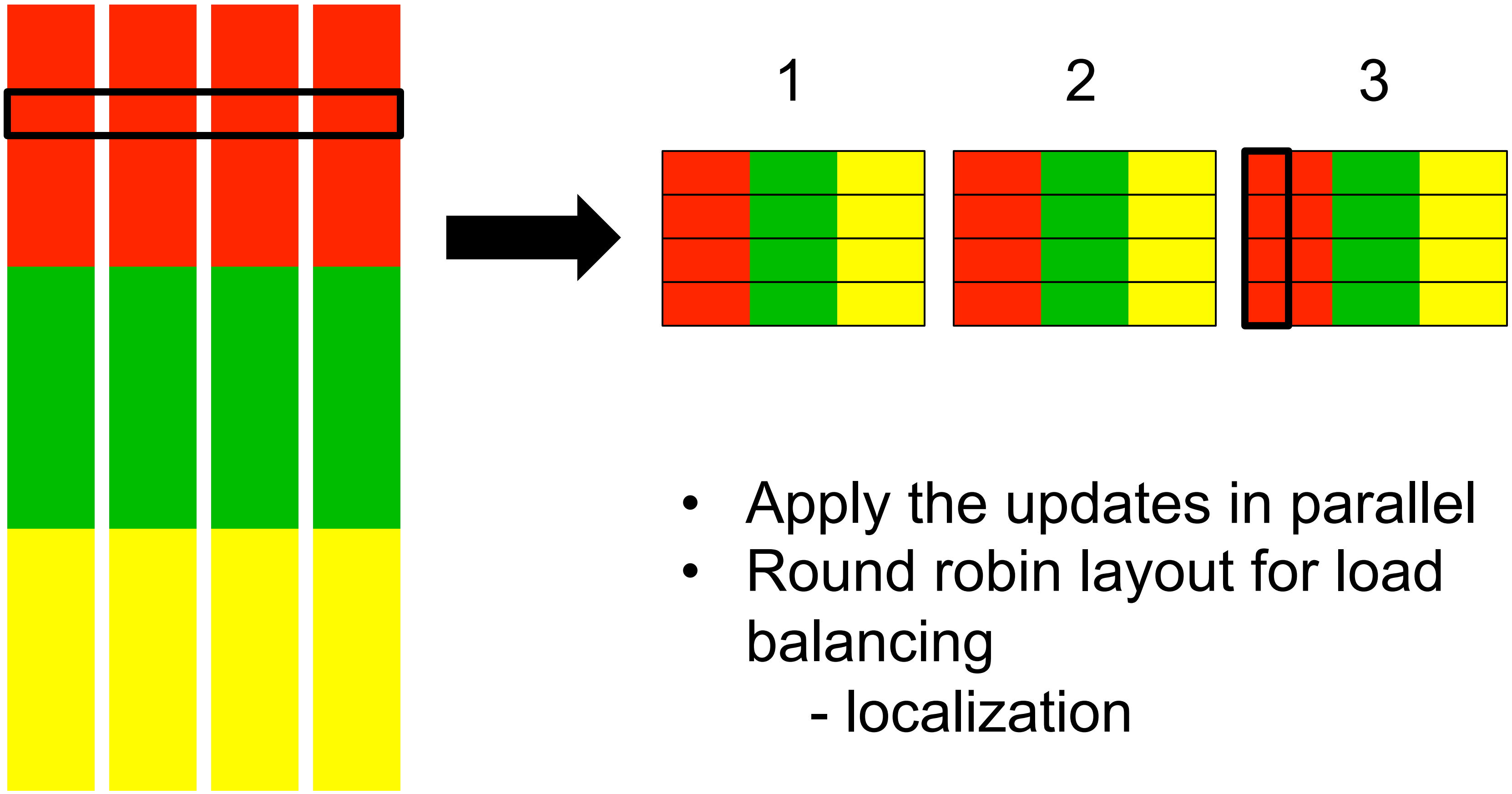
- Apply the updates in parallel
- Round robin layout for load balancing
  - localization

Assimilation



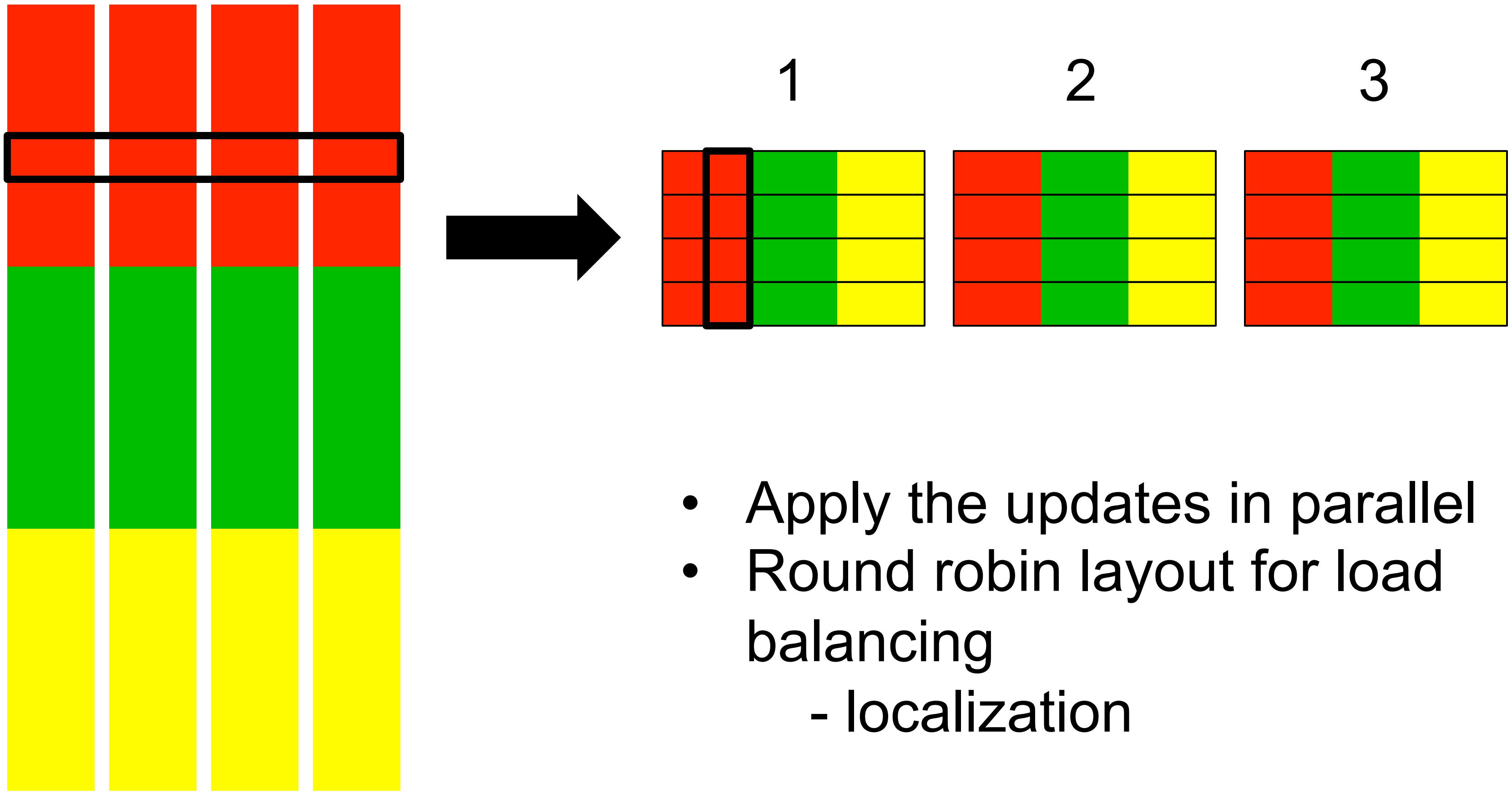
- Apply the updates in parallel
- Round robin layout for load balancing
  - localization

Assimilation



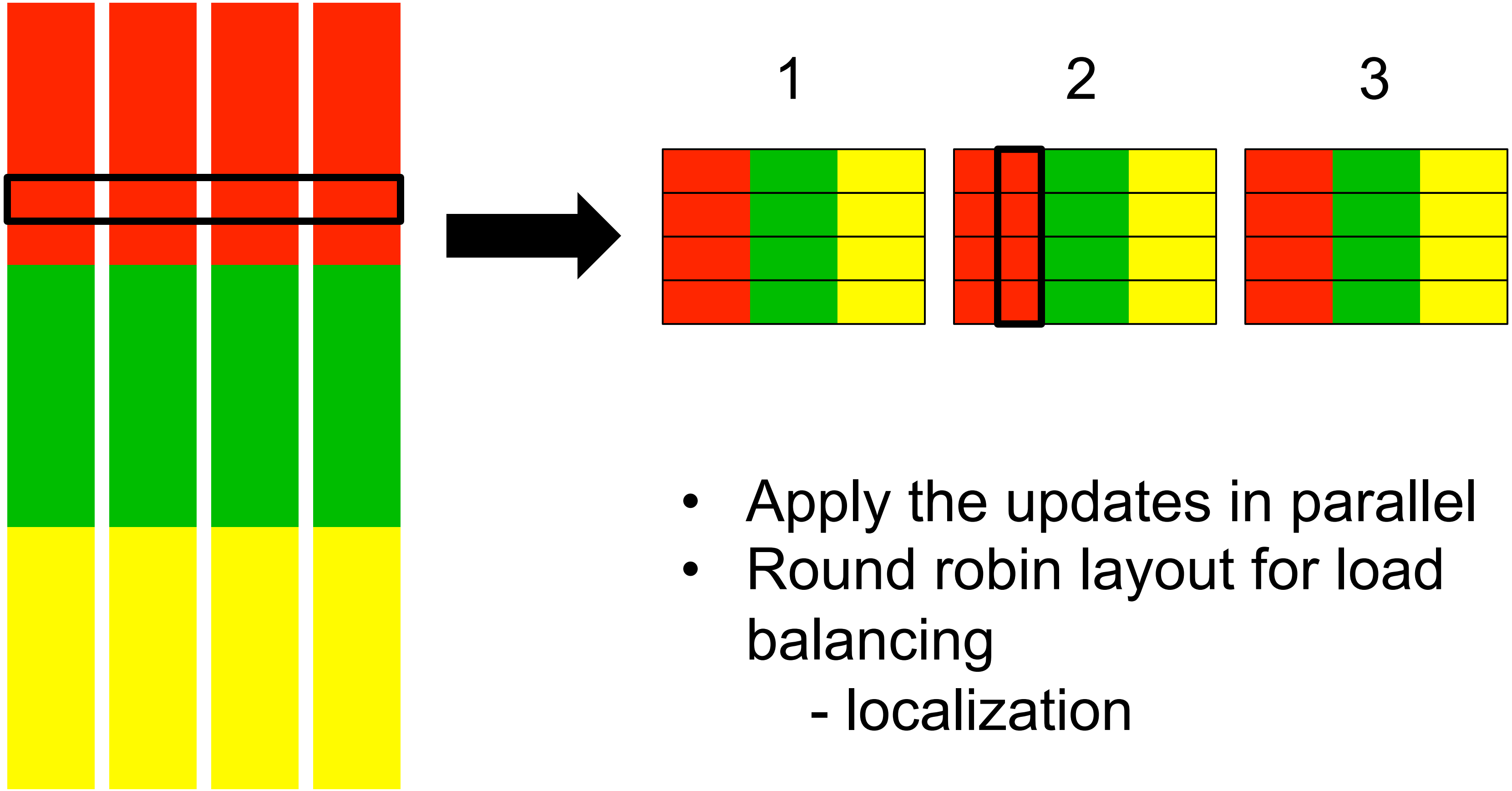
- Apply the updates in parallel
- Round robin layout for load balancing
  - localization

Assimilation



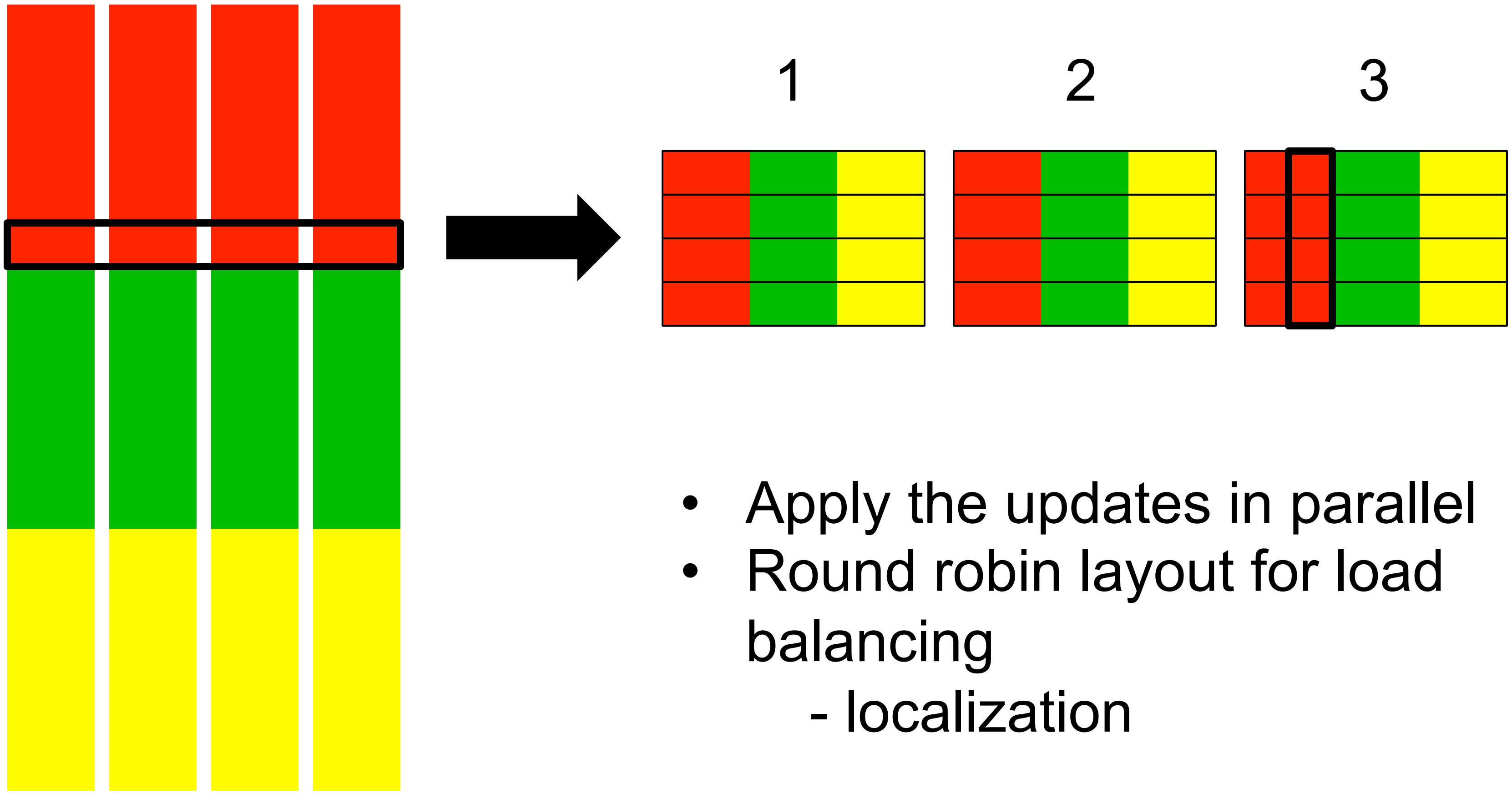
- Apply the updates in parallel
- Round robin layout for load balancing
  - localization

Assimilation



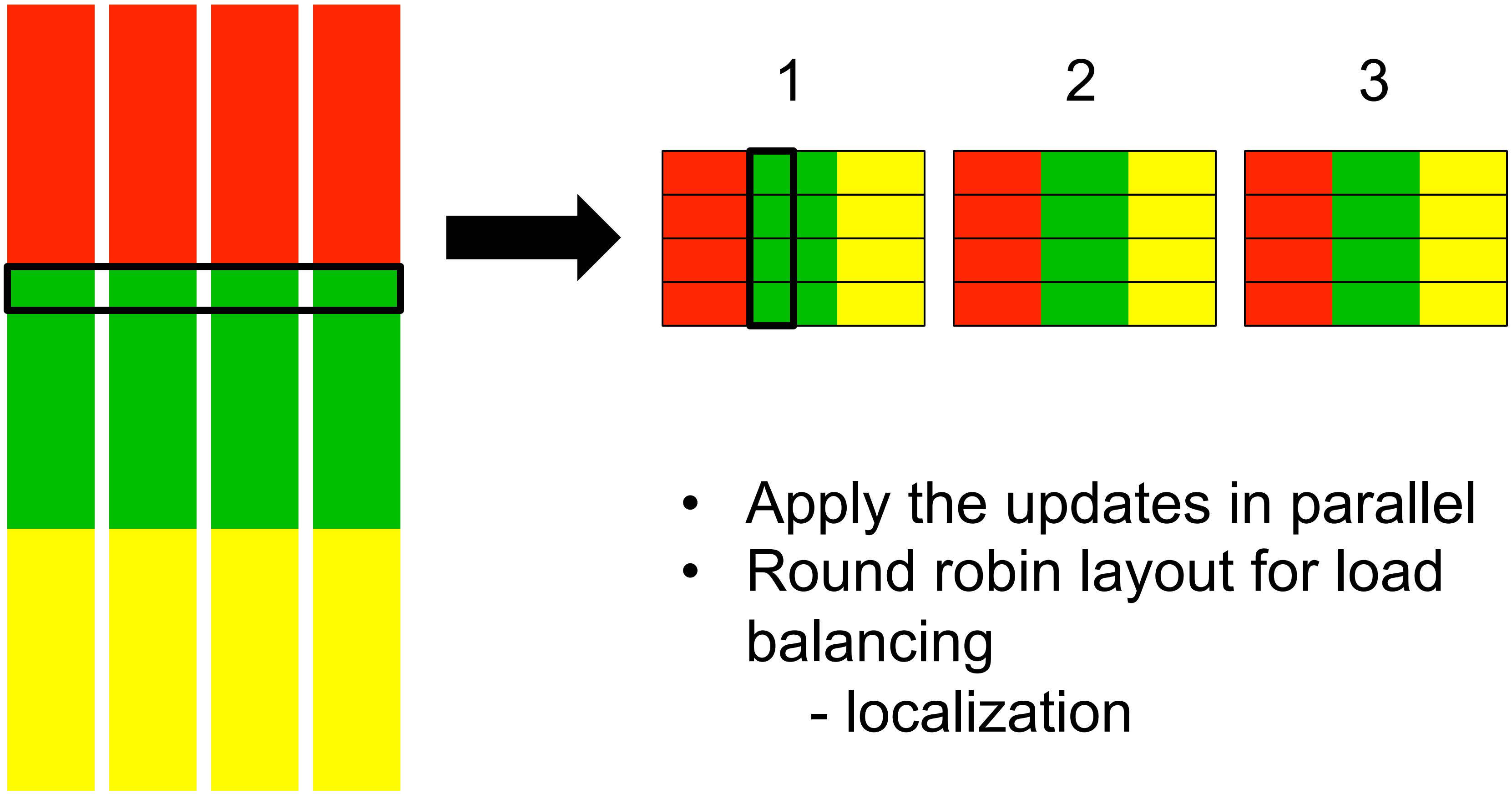
- Apply the updates in parallel
- Round robin layout for load balancing
  - localization

Assimilation



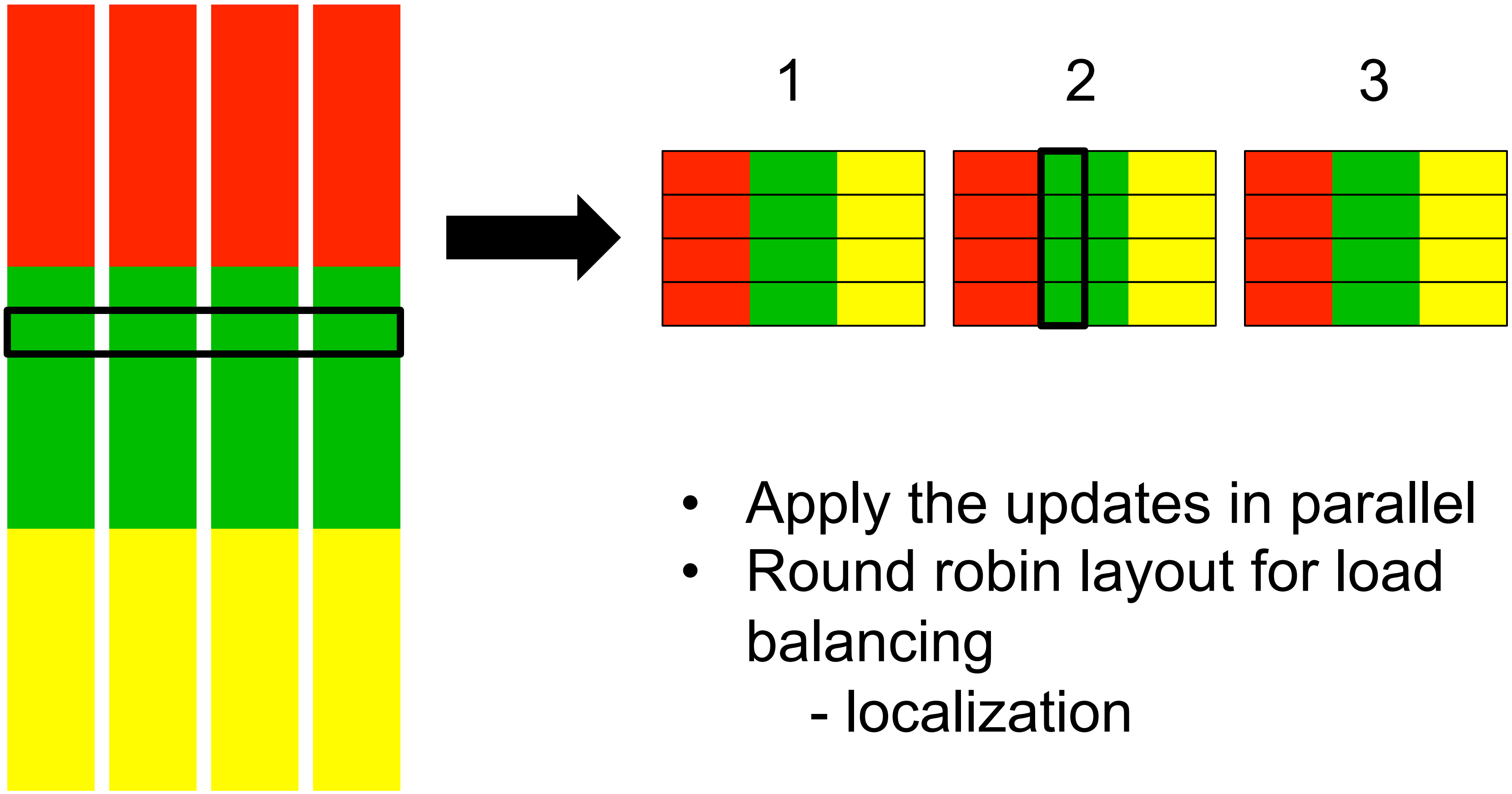
- Apply the updates in parallel
- Round robin layout for load balancing
  - localization

Assimilation



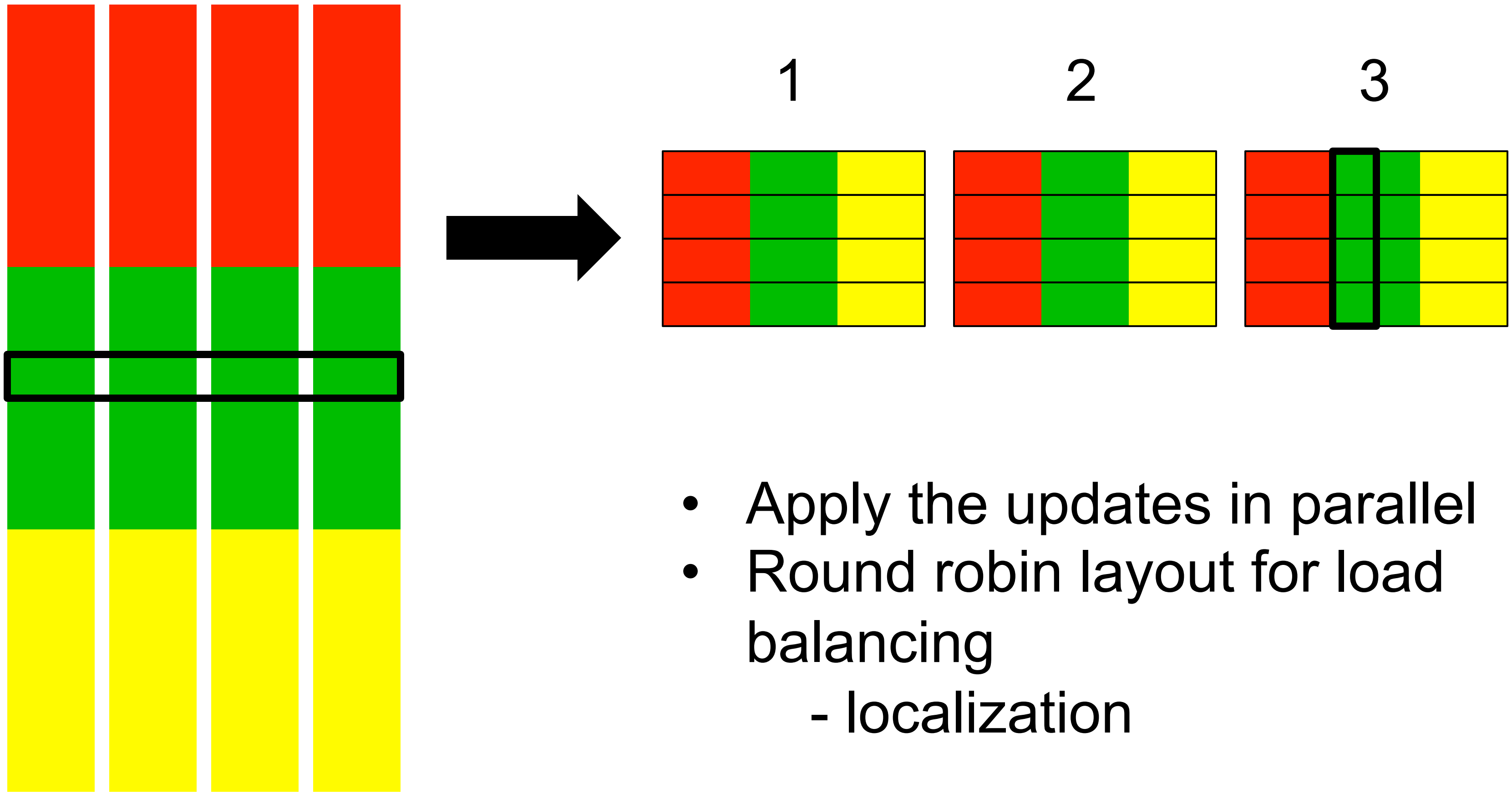
- Apply the updates in parallel
- Round robin layout for load balancing
  - localization

Assimilation



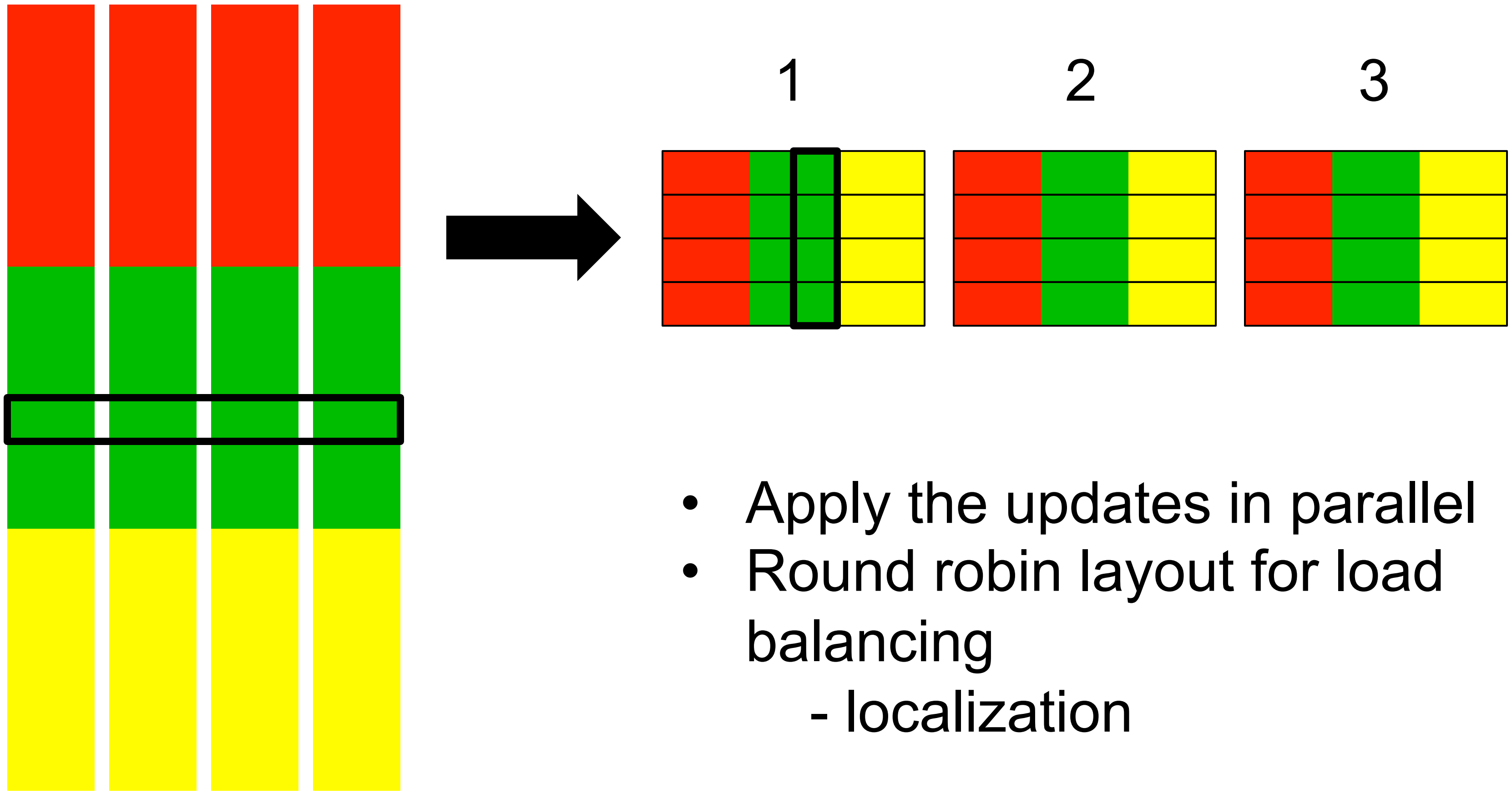
- Apply the updates in parallel
- Round robin layout for load balancing
  - localization

Assimilation



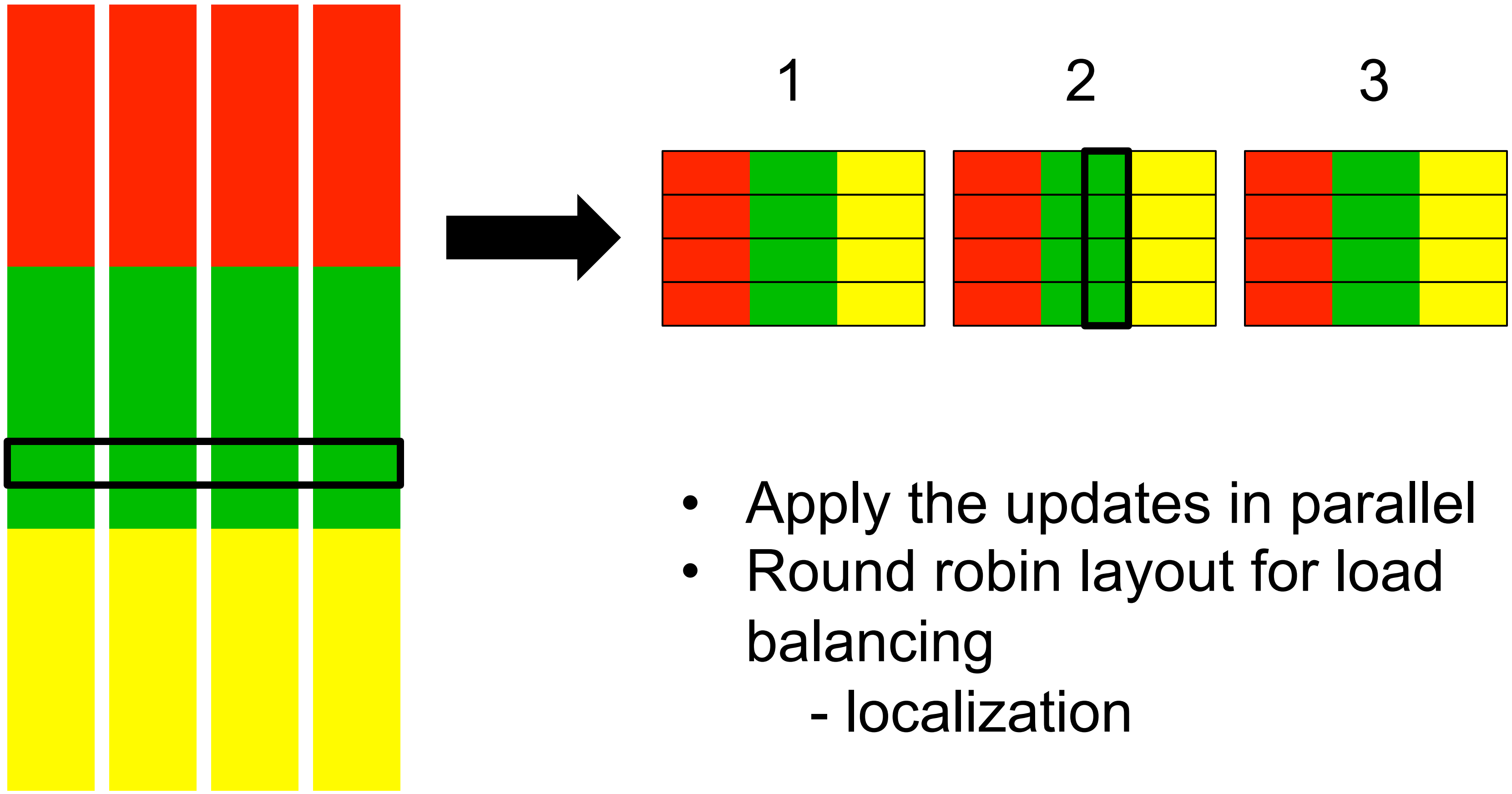
- Apply the updates in parallel
- Round robin layout for load balancing
    - localization

Assimilation



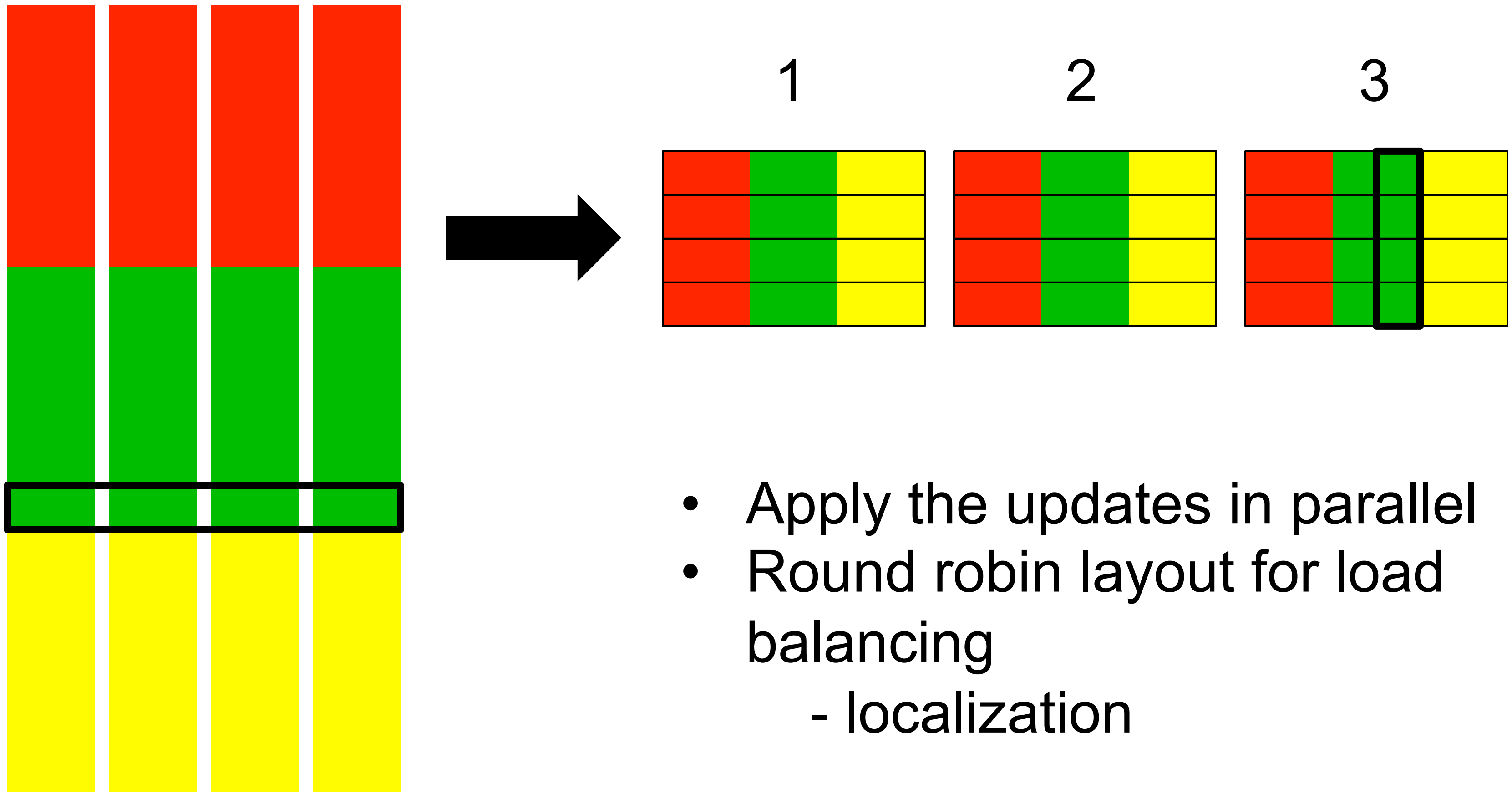
- Apply the updates in parallel
- Round robin layout for load balancing
    - localization

Assimilation



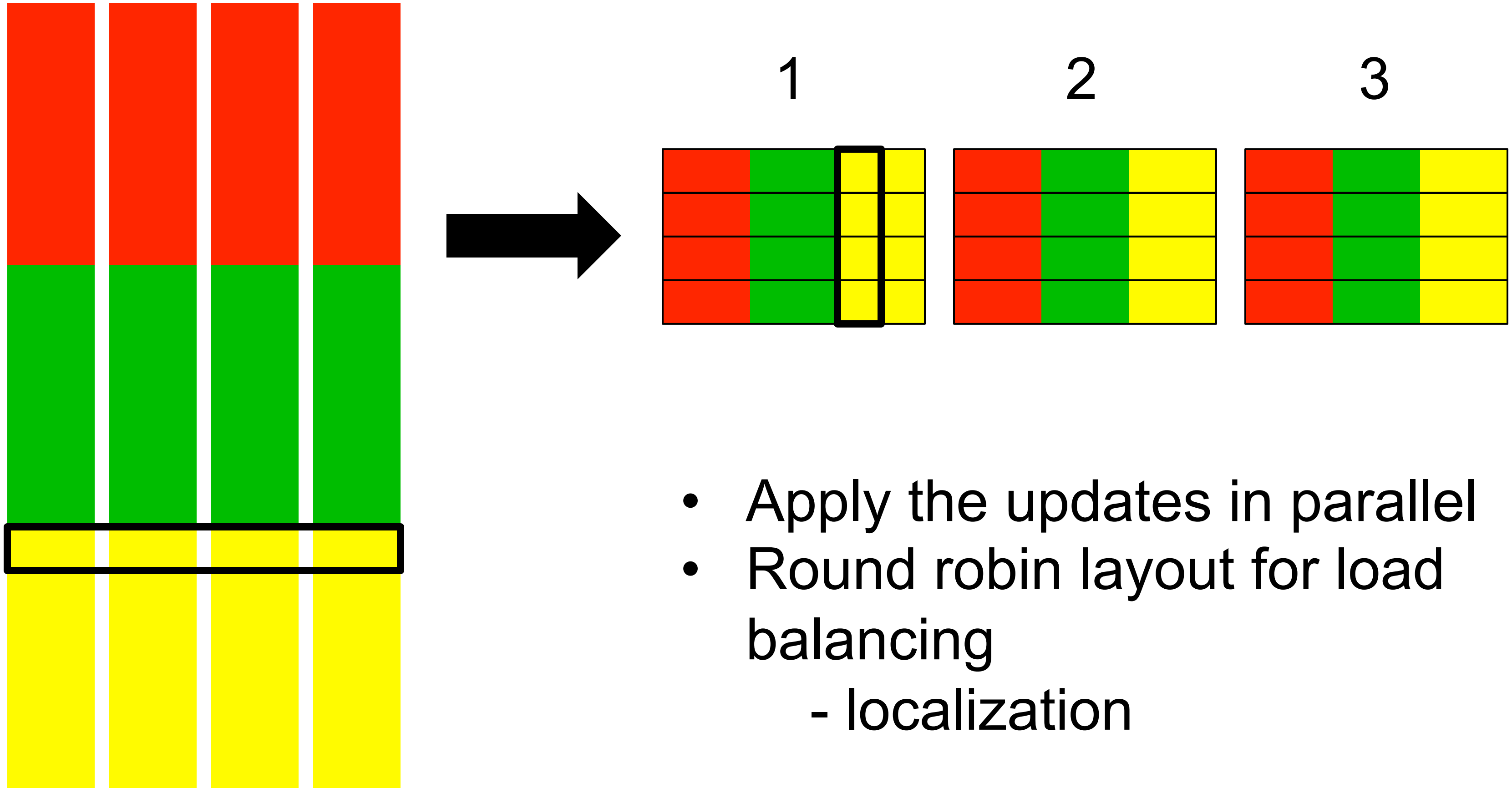
- Apply the updates in parallel
- Round robin layout for load balancing
    - localization

Assimilation



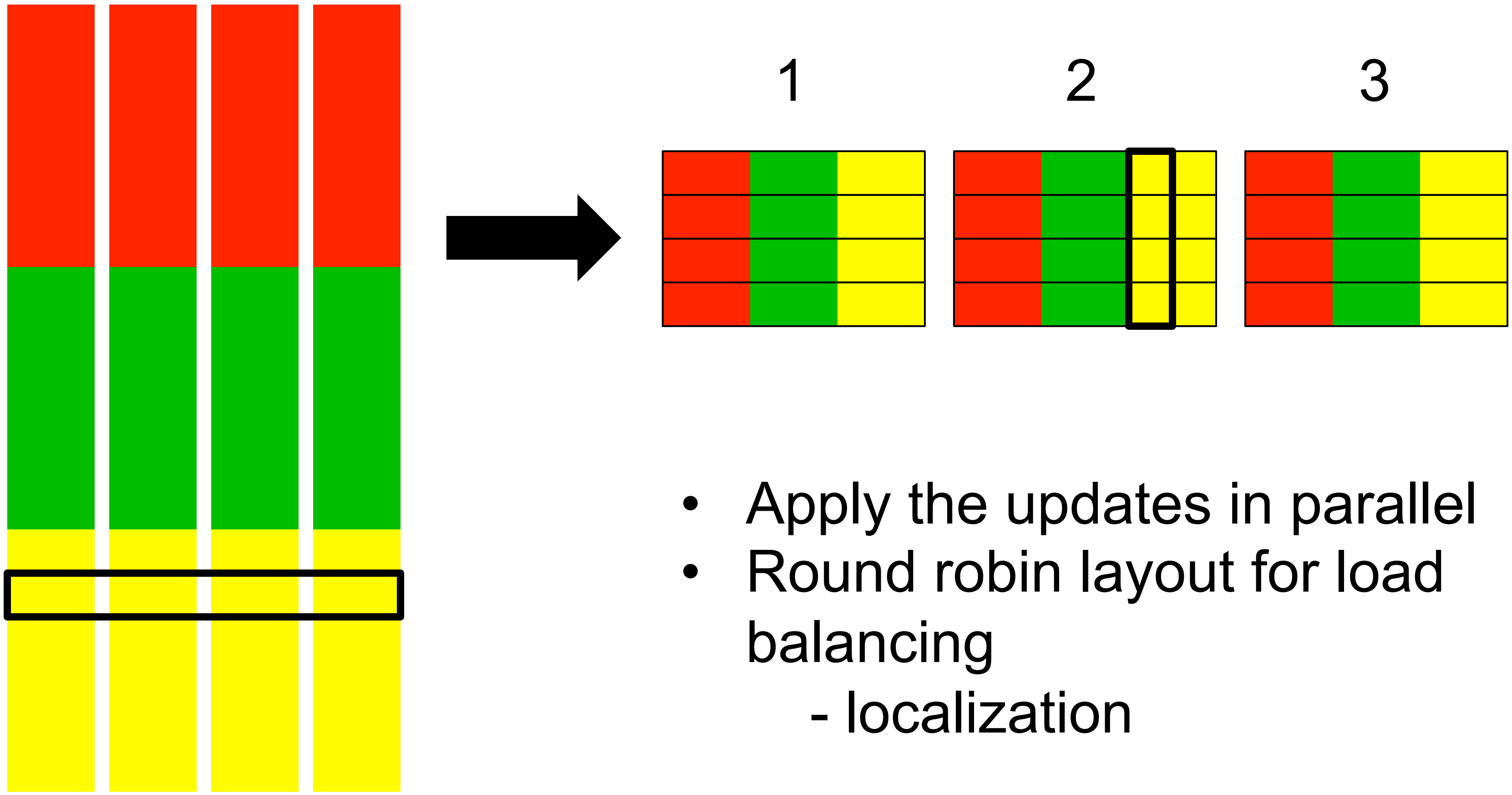
- Apply the updates in parallel
- Round robin layout for load balancing
  - localization

Assimilation



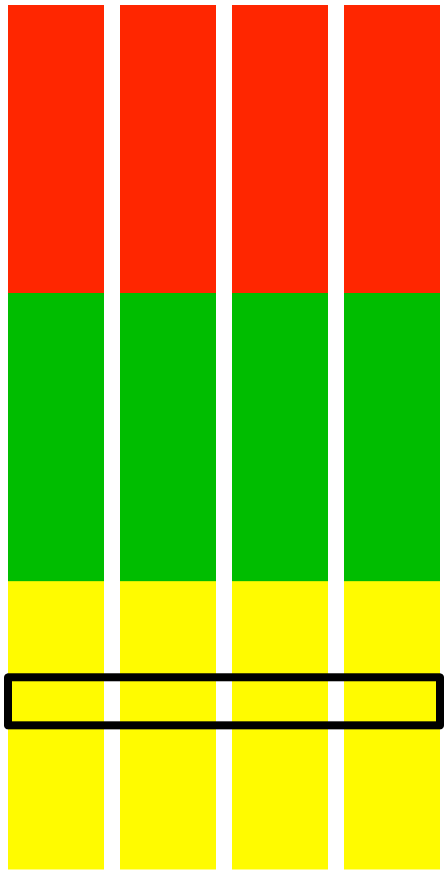
- Apply the updates in parallel
- Round robin layout for load balancing
  - localization

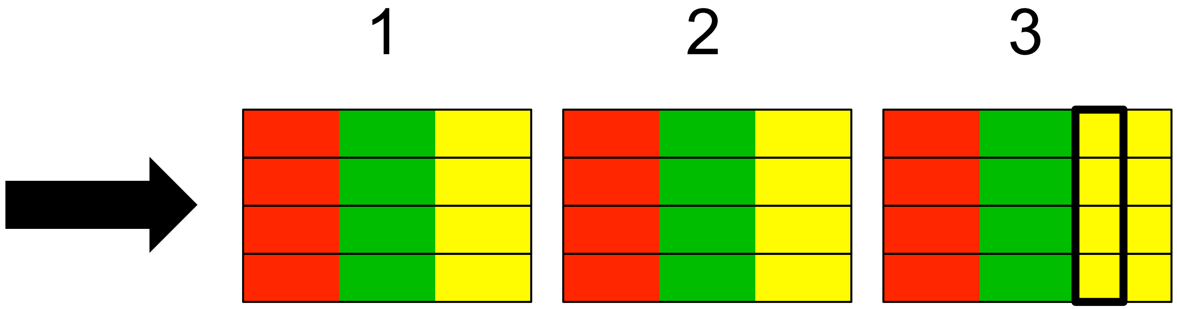
Assimilation



- Apply the updates in parallel
- Round robin layout for load balancing
  - localization

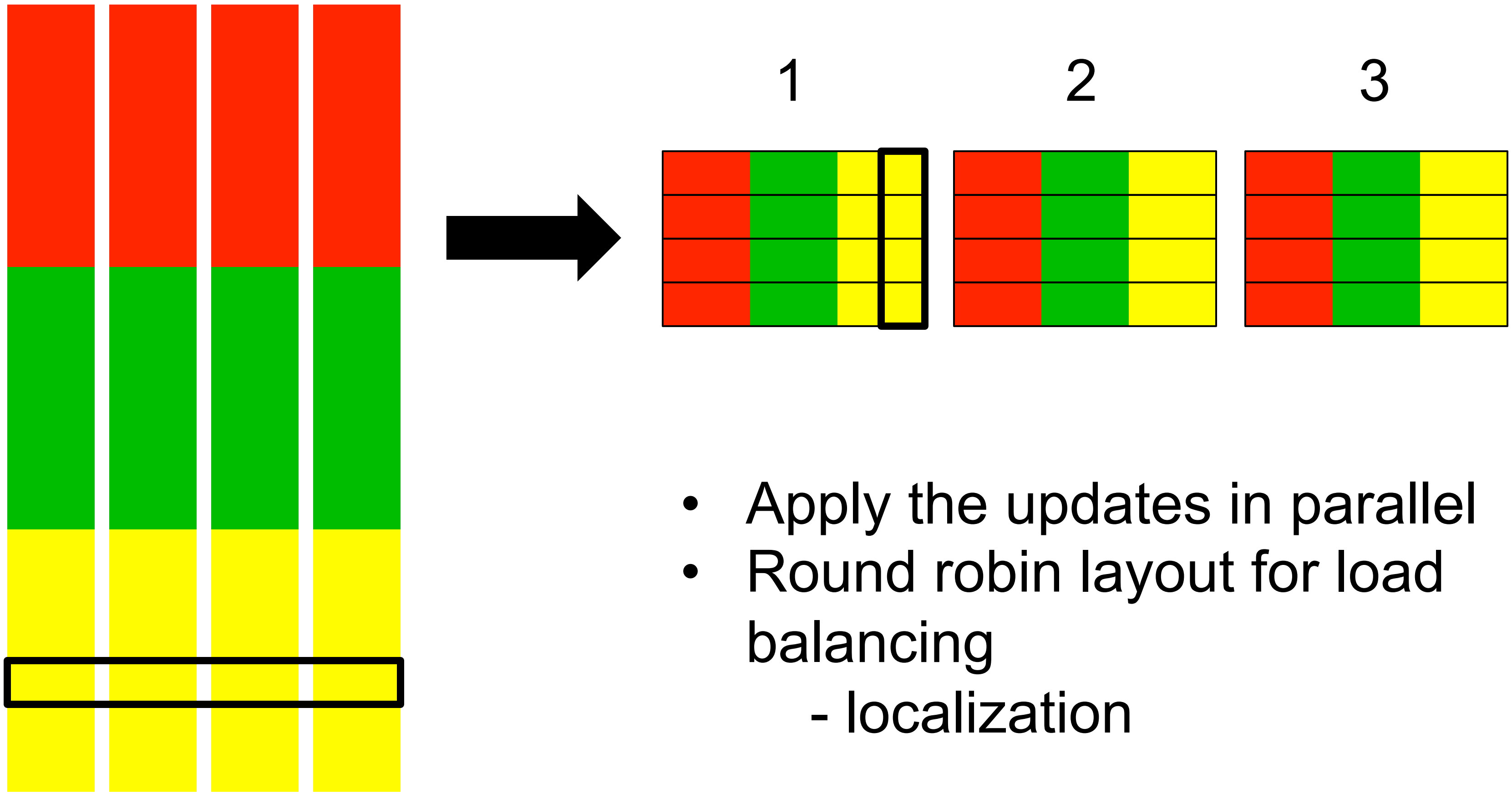
Assimilation





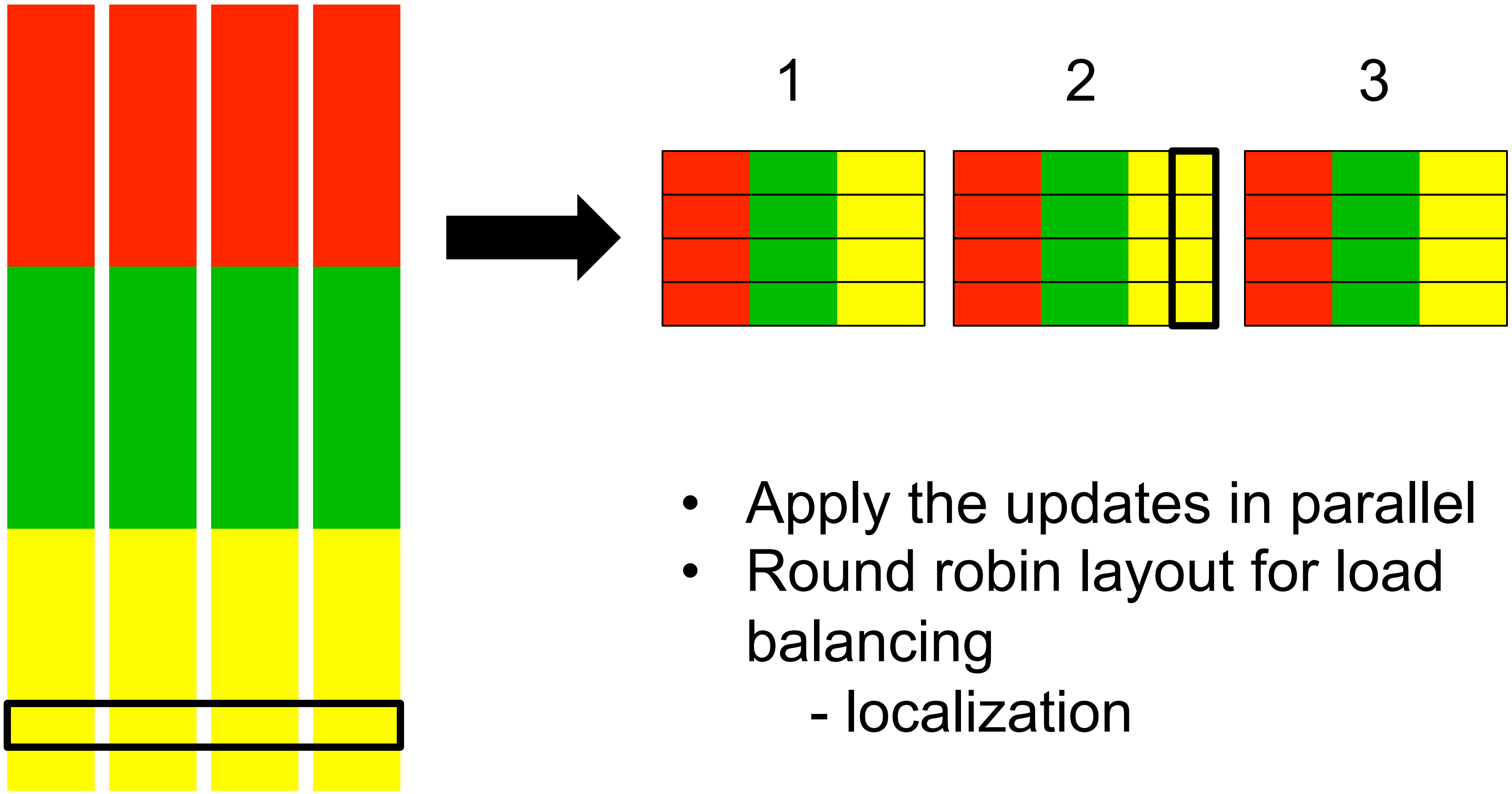
- Apply the updates in parallel
- Round robin layout for load balancing
  - localization

Assimilation



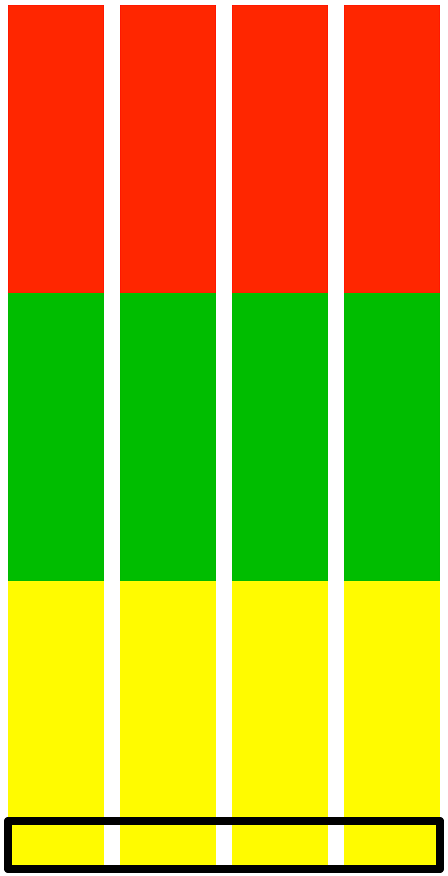
- Apply the updates in parallel
- Round robin layout for load balancing
  - localization

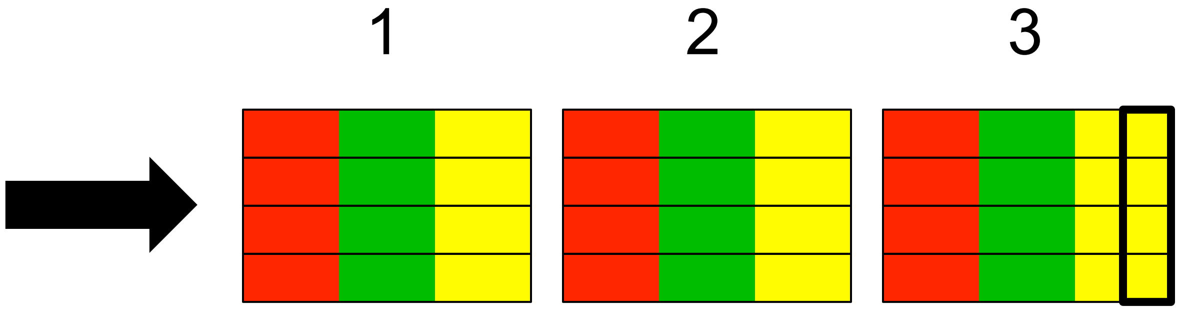
Assimilation



- Apply the updates in parallel
- Round robin layout for load balancing
    - localization

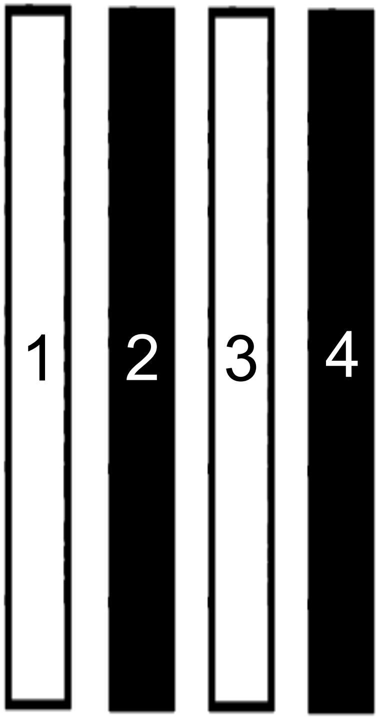
Assimilation



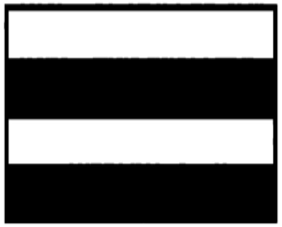


- Apply the updates in parallel
- Round robin layout for load balancing
  - localization

Data decompositions

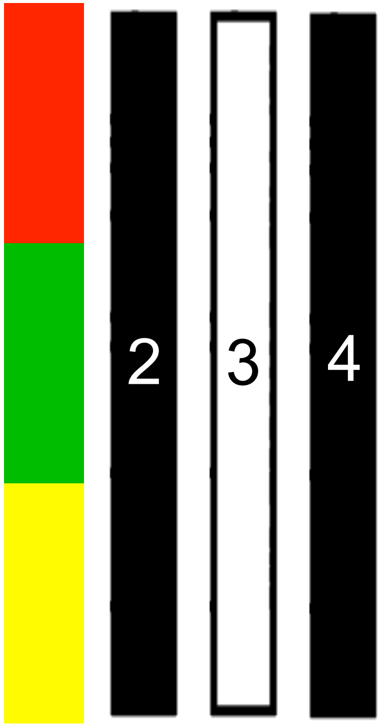


Whole model state available to a single processor

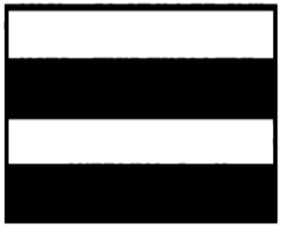


All copies of some variables available to a single processor

Data decompositions

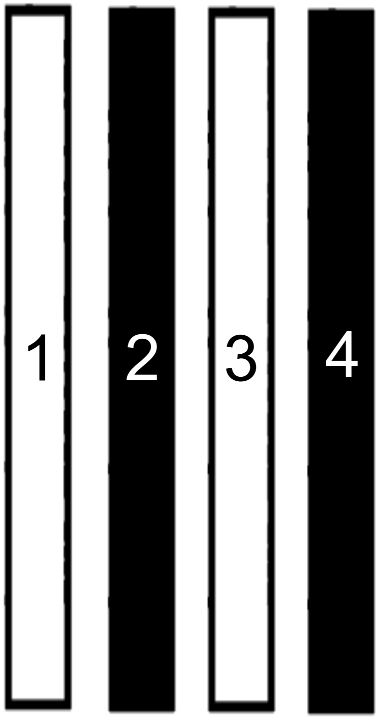


Whole model state available to a single processor

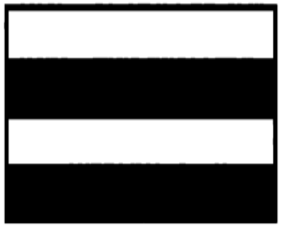


All copies of some variables available to a single processor

Data decompositions



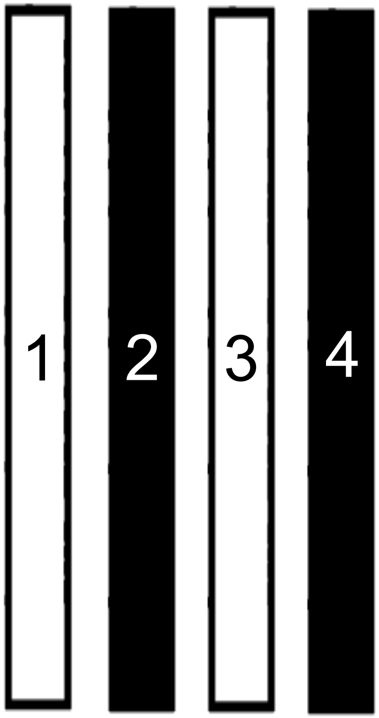
Whole model state available to a single processor



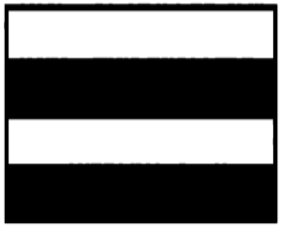
All copies of some variables available to a single processor

Why do we need to change anything?

What does DART look like in memory?

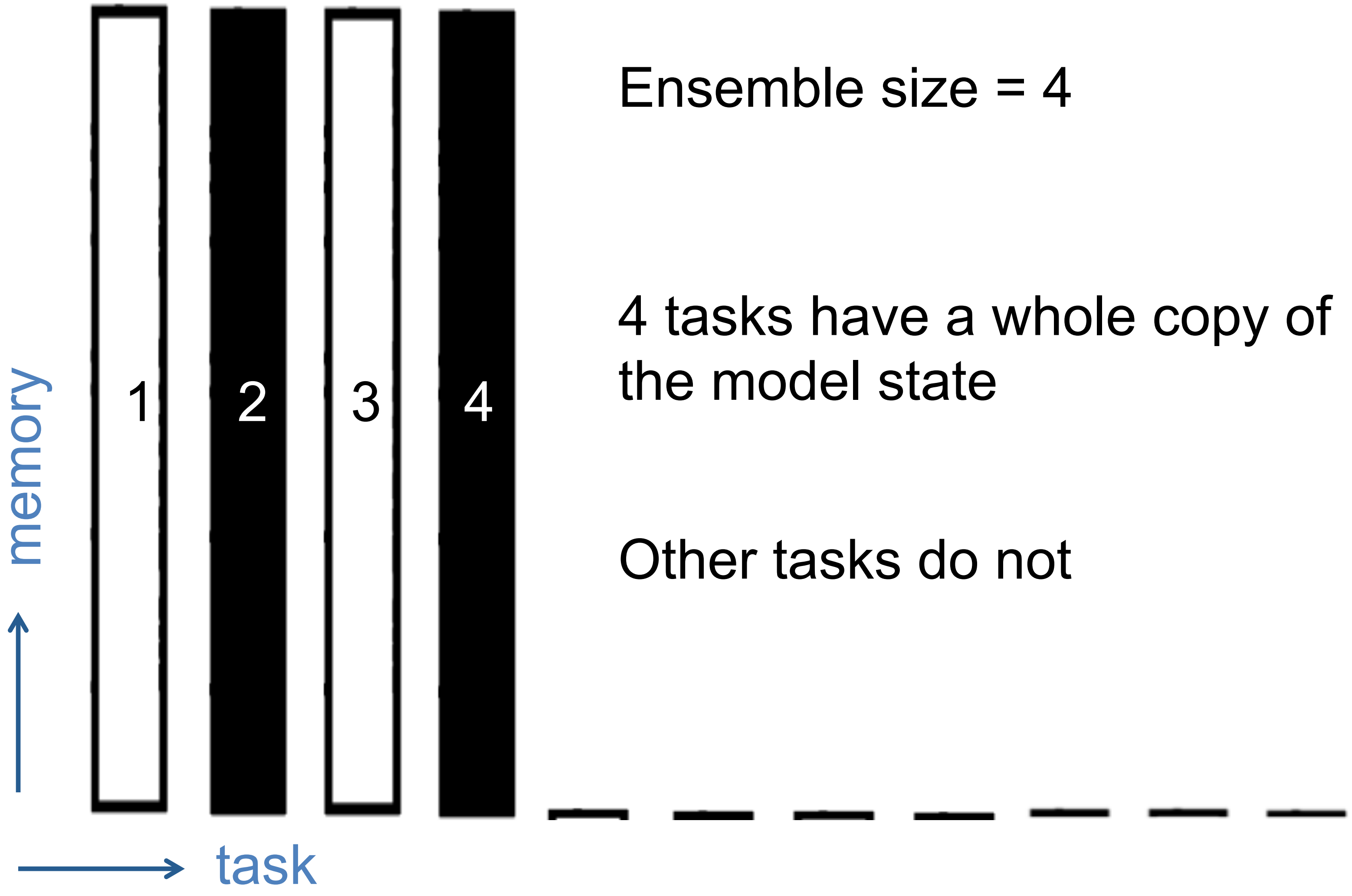


Whole model state available to a single processor



All copies of some variables available to a single processor

What does DART look like in memory?



Ensemble size = 4

4 tasks have a whole copy of the model state

Other tasks do not

Why do we use this decomposition?

Calculation of the forward operator

Why do we use this decomposition?

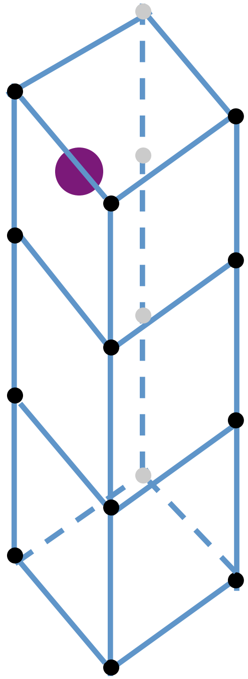
Calculation of the forward operator

What the model thinks the observation should be

Why do we use this decomposition?

Calculation of the forward operator

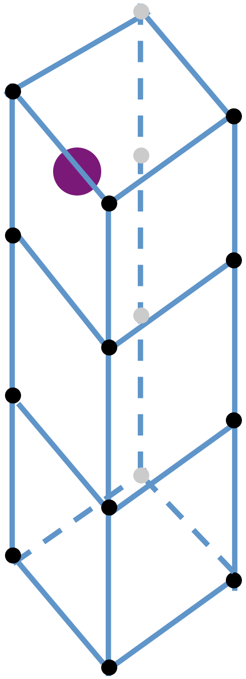
What the model thinks the observation should be

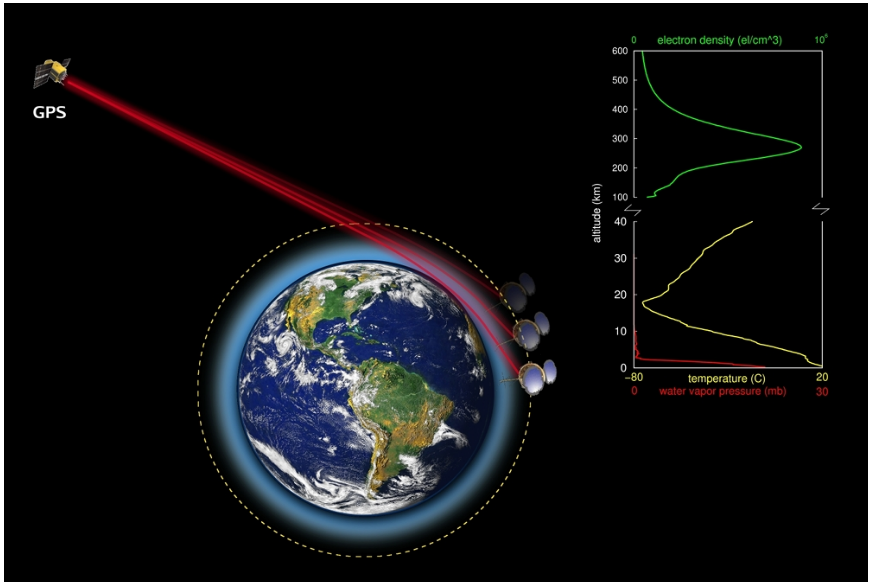


Why do we use this decomposition?

Calculation of the forward operator

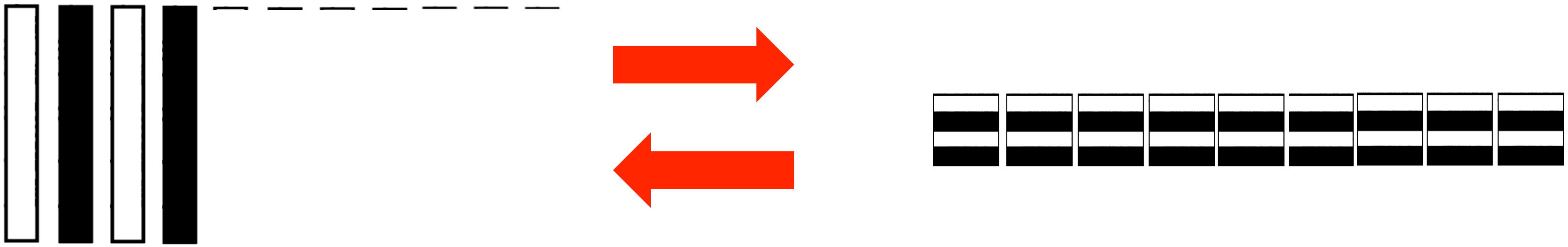
What the model thinks the observation should be





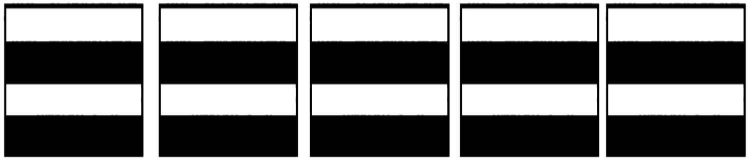
Limitations of having these two decompositions:

- Hard minimum on calculation time
- Hard maximum on model size
- You have to move all your data



Idea:

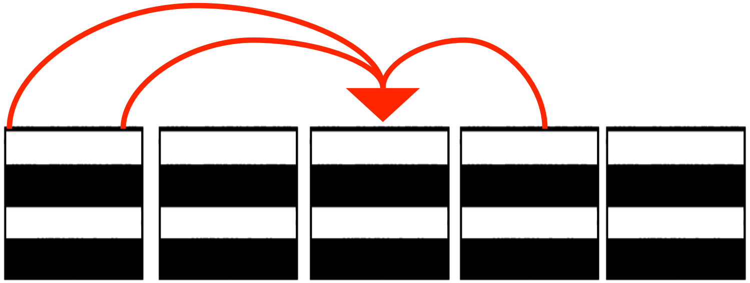
Only use the assimilation decomposition



Idea:

Only use the assimilation decomposition

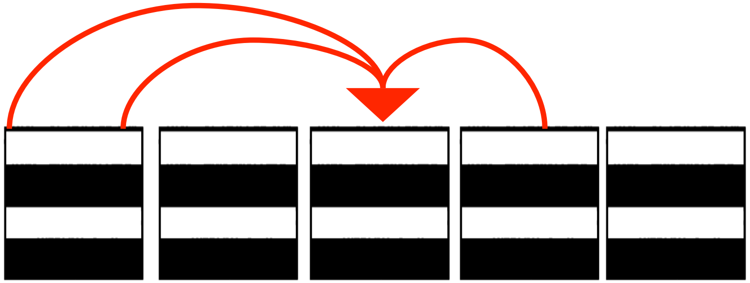
Use **one sided communication** to grab state elements when needed



Idea:

Only use the assimilation decomposition

Use **one sided communication** to grab state elements when needed



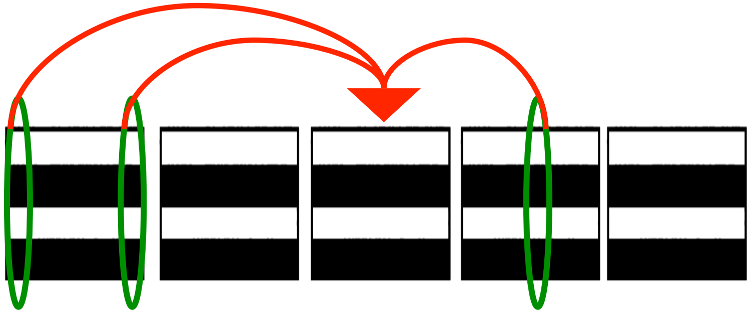
Reduce data movement

Removes hard memory limit

Idea:

Only use the assimilation decomposition

Use **one sided communication** to grab state elements when needed



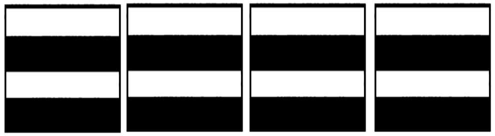
Reduce data movement

Removes hard memory limit

Vectorization of forward operator calculations

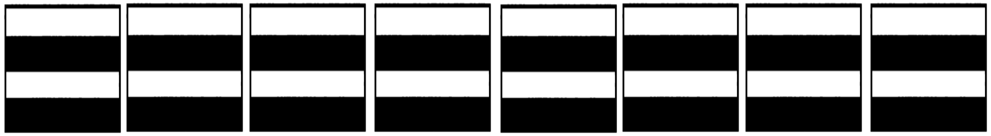
More scalable forward operator

Memory



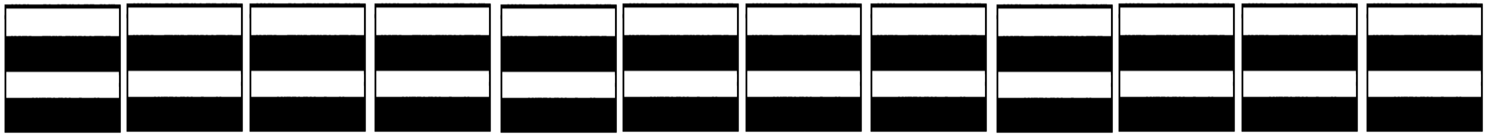
More scalable forward operator

Memory



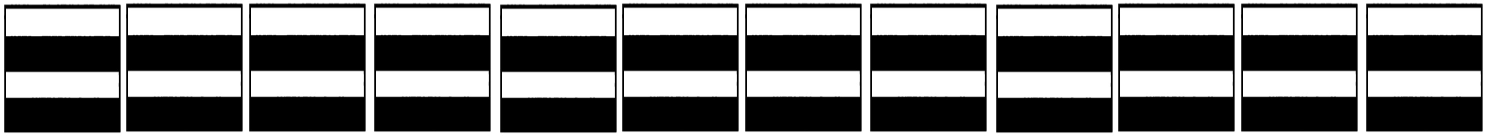
More scalable forward operator

Memory

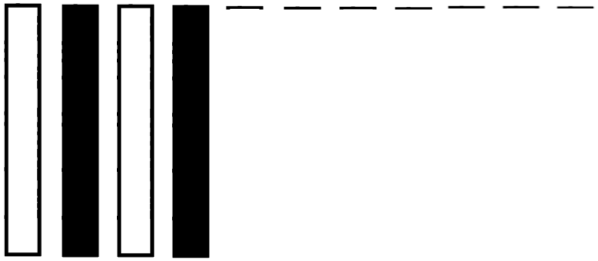


More scalable forward operator

Memory



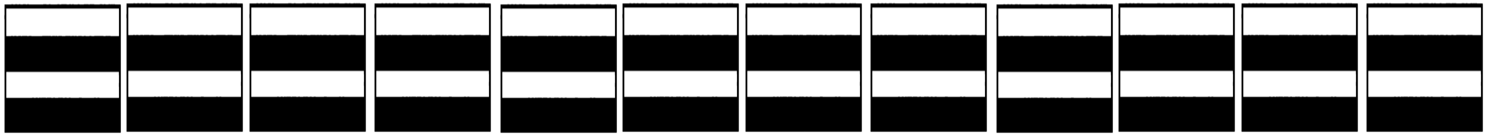
Calculation



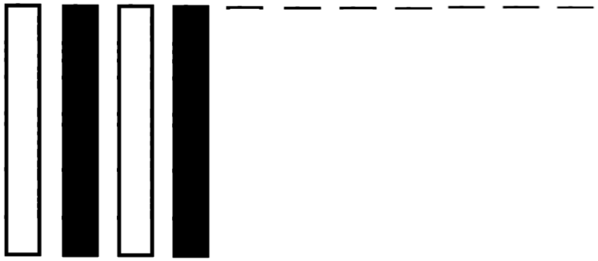
4 tasks doing all observations for 1 copy

More scalable forward operator

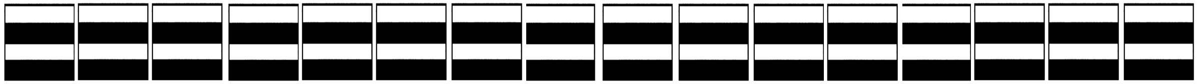
Memory



Calculation

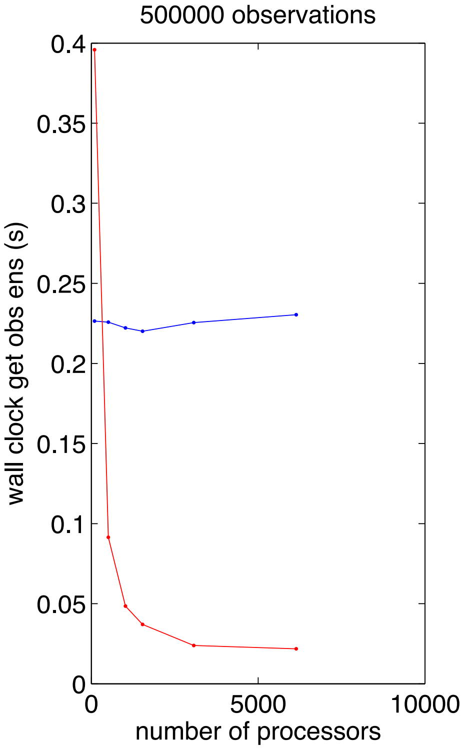


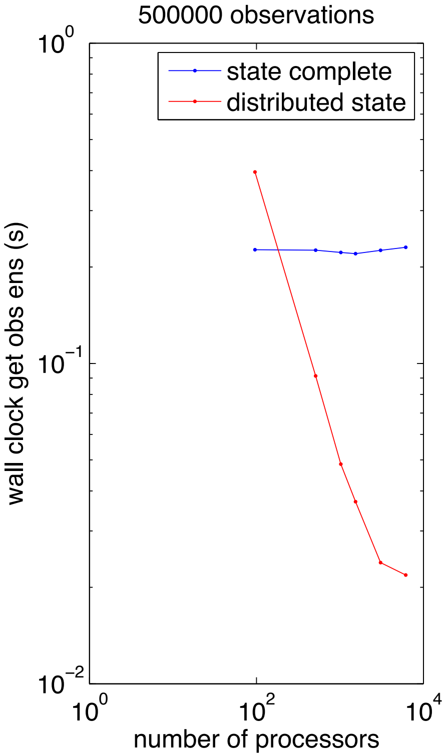
4 tasks doing all observations for 1 copy

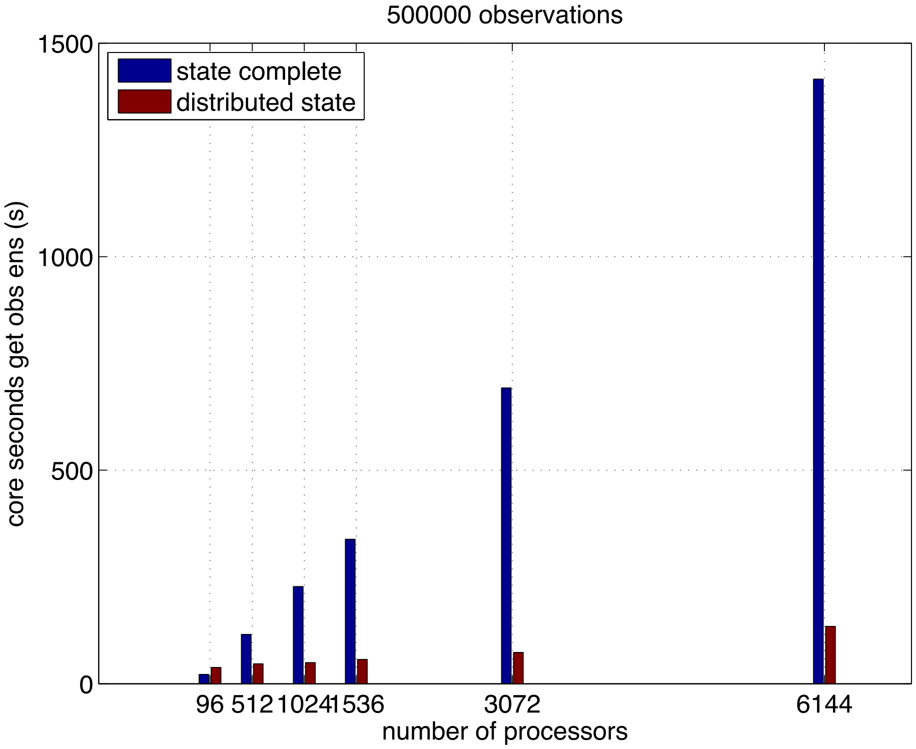


Lots of tasks doing some observations for all copies

Lorenz_96 forward operator





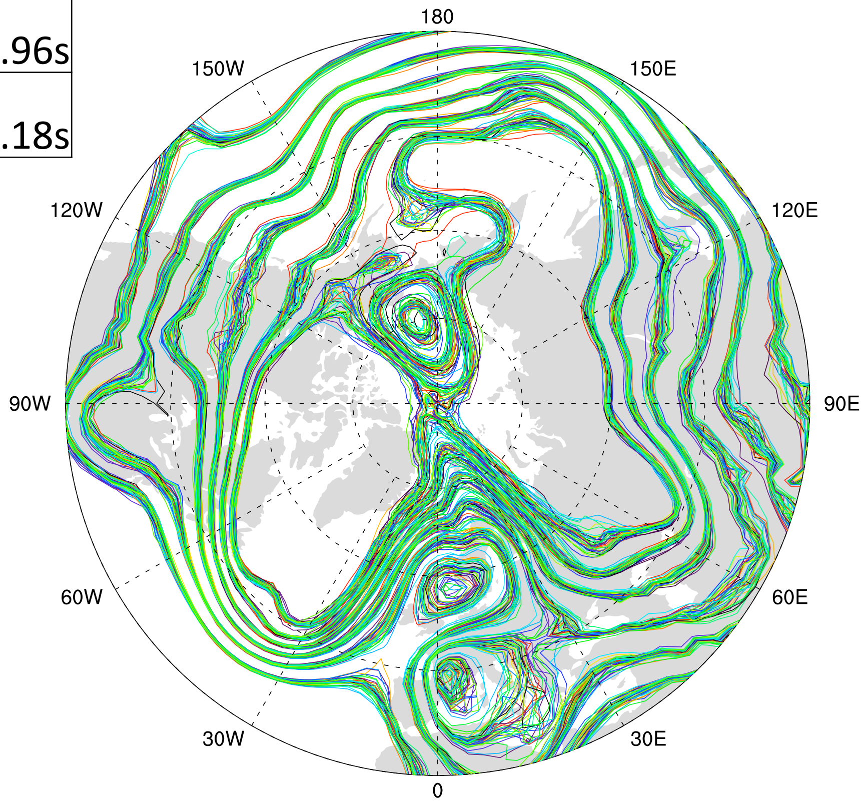


wall clock

core seconds

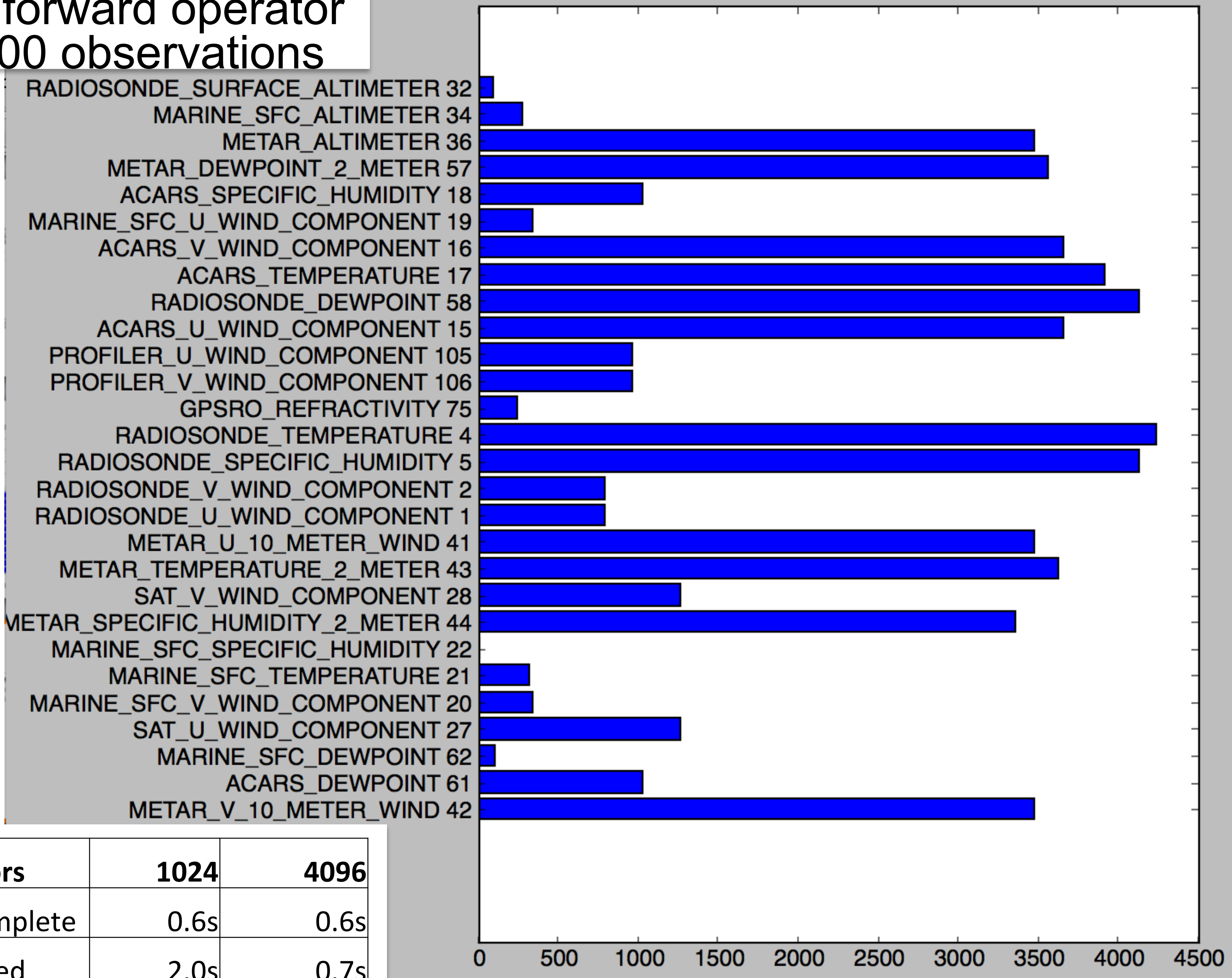
CAM FV forward operator
Specific humidity only : 23 090 observations

processors	**512**	**4096**
state complete	1.01s	0.96s
distributed state	0.73s	0.18s



CONTOUR FROM 5200 TO 5700 BY 100

WRF forward operator
54, 400 observations

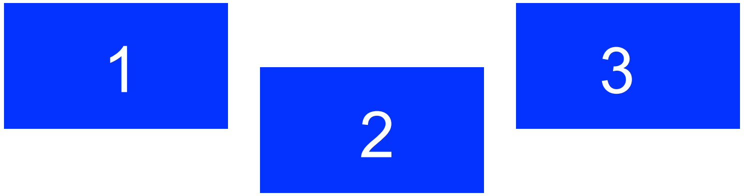


processors	**1024**	**4096**
state complete	0.6s	0.6s
distributed	2.0s	0.7s

IO

IO

Models do not run ensemble complete



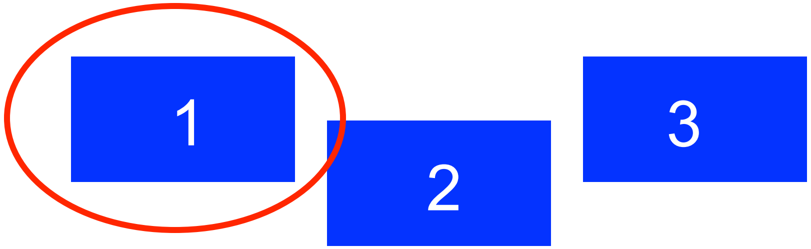
1

2

3

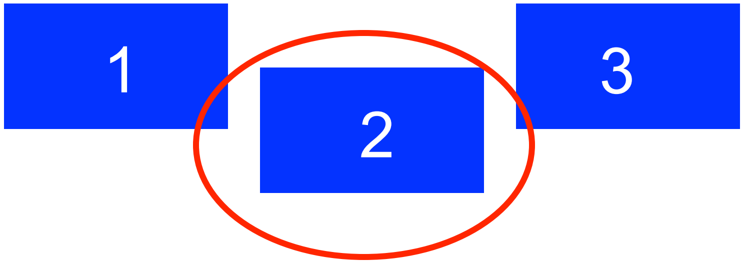
IO

Models do not run ensemble complete



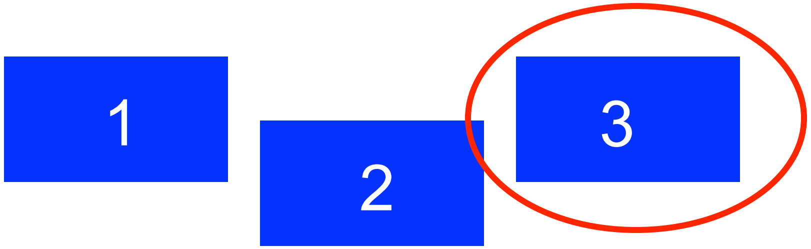
IO

Models do not run ensemble complete



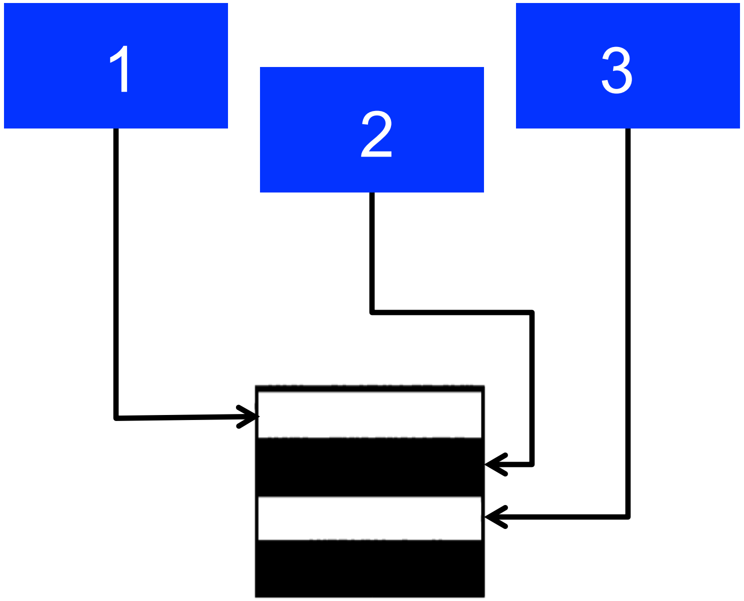
IO

Models do not run ensemble complete



IO

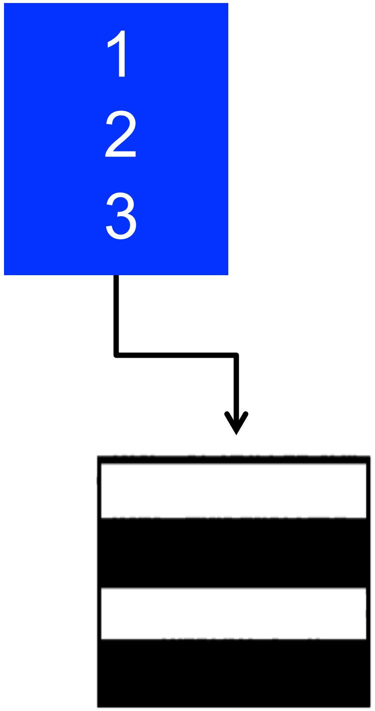
Models do not run ensemble complete



You have to move data from the model to DART

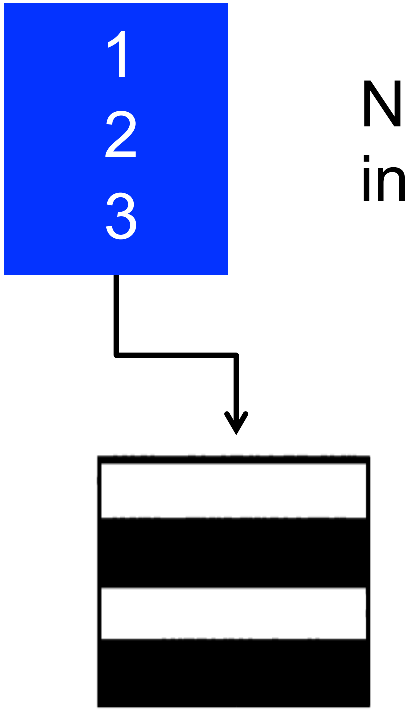
IO

Ideally:



IO

Ideally:

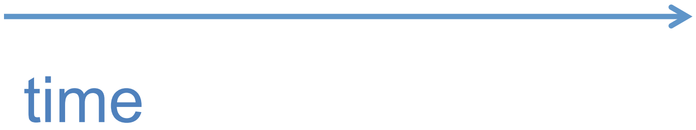


Never looks like this in memory

IO

All DART requires is that there are multiple model forecasts

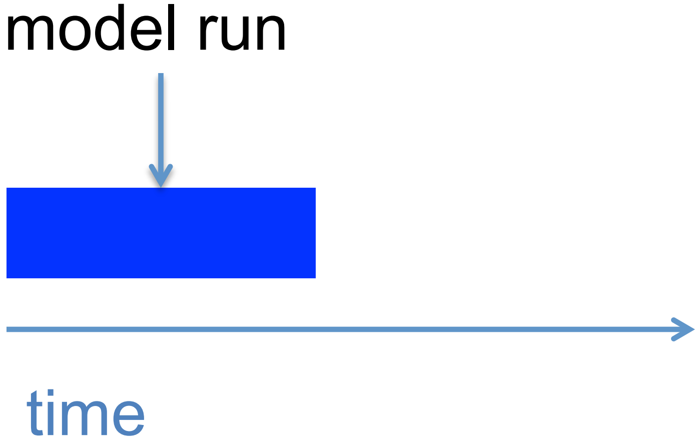
IO



time

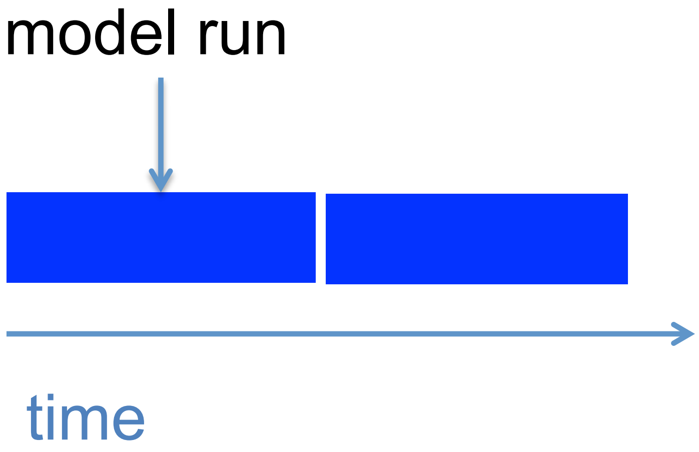
Multiple model forecasts to create the ensemble

IO



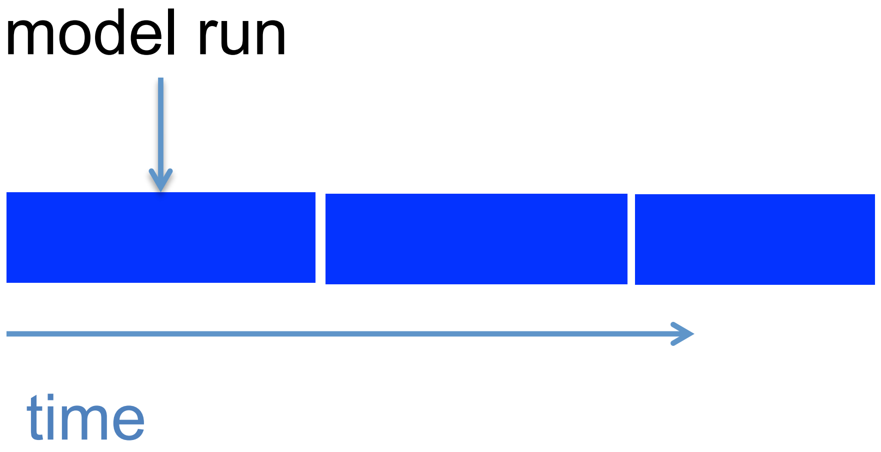
Multiple model forecasts to create the ensemble

IO



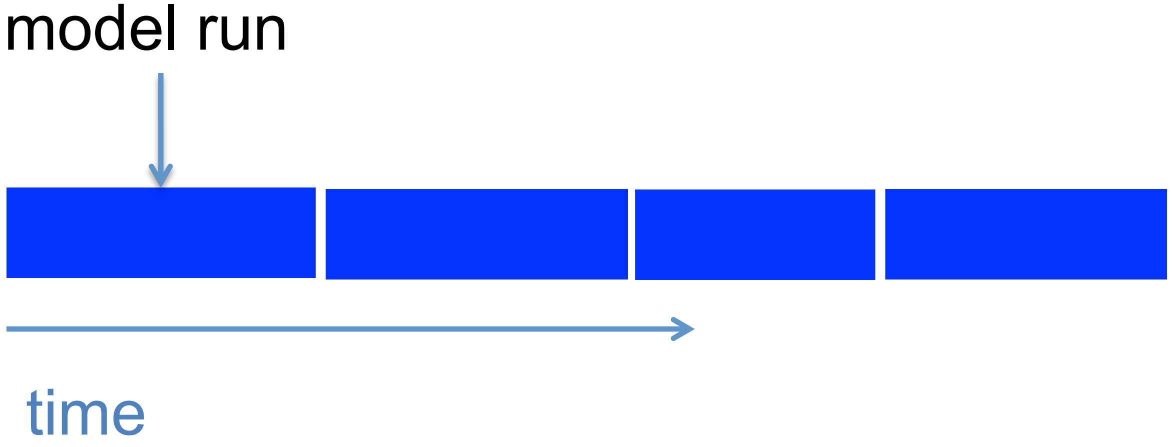
Multiple model forecasts to create the ensemble

IO



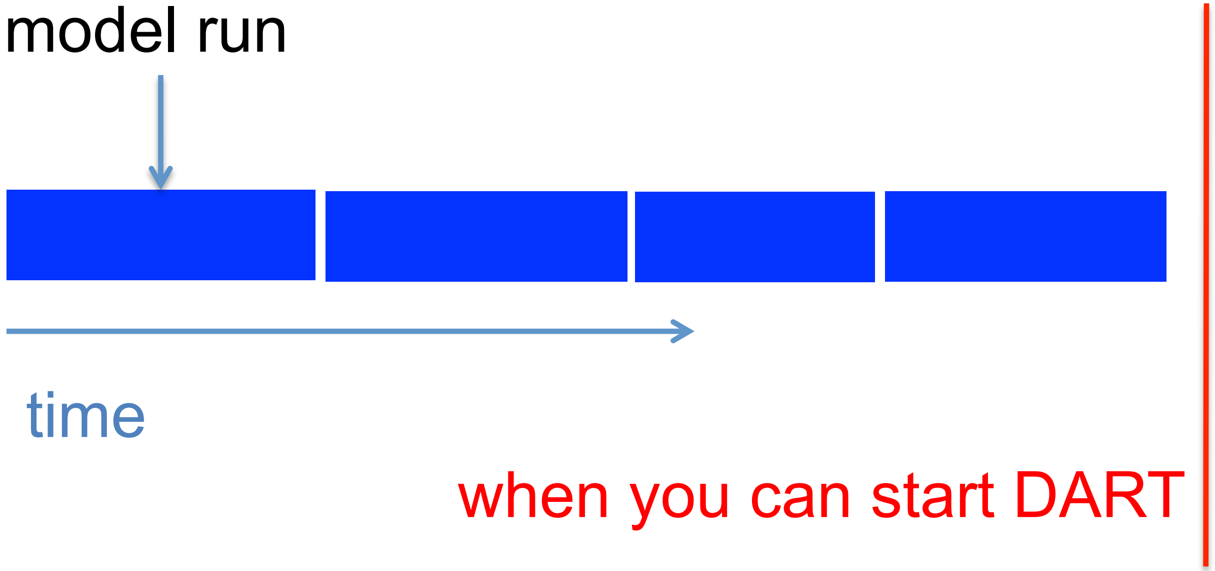
Multiple model forecasts to create the ensemble

IO



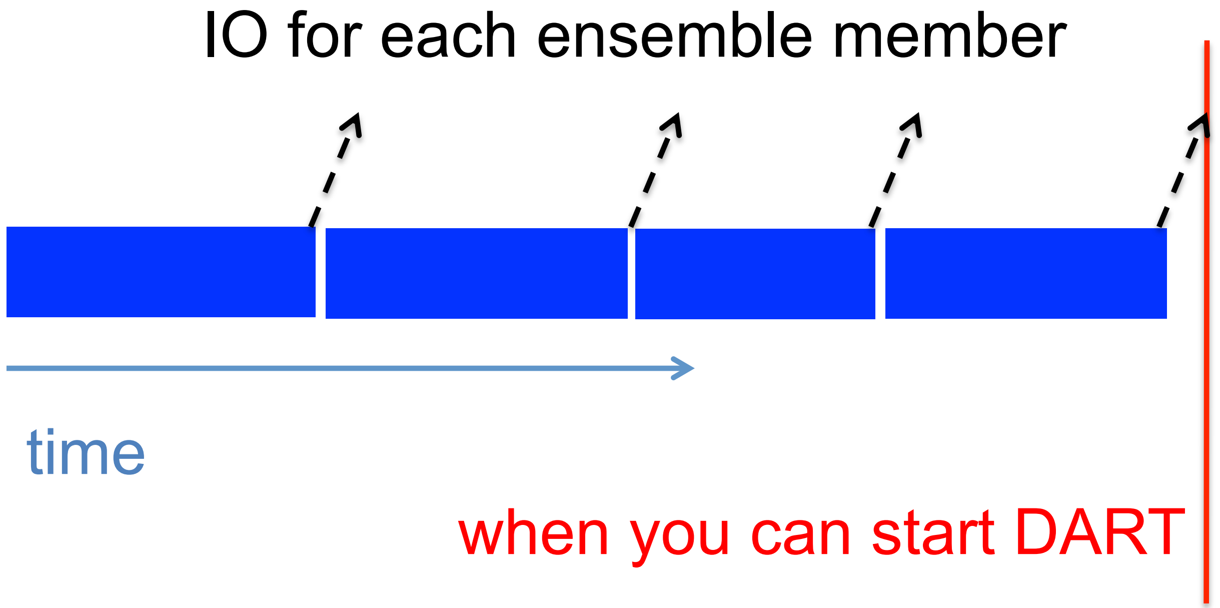
Multiple model forecasts to create the ensemble

IO



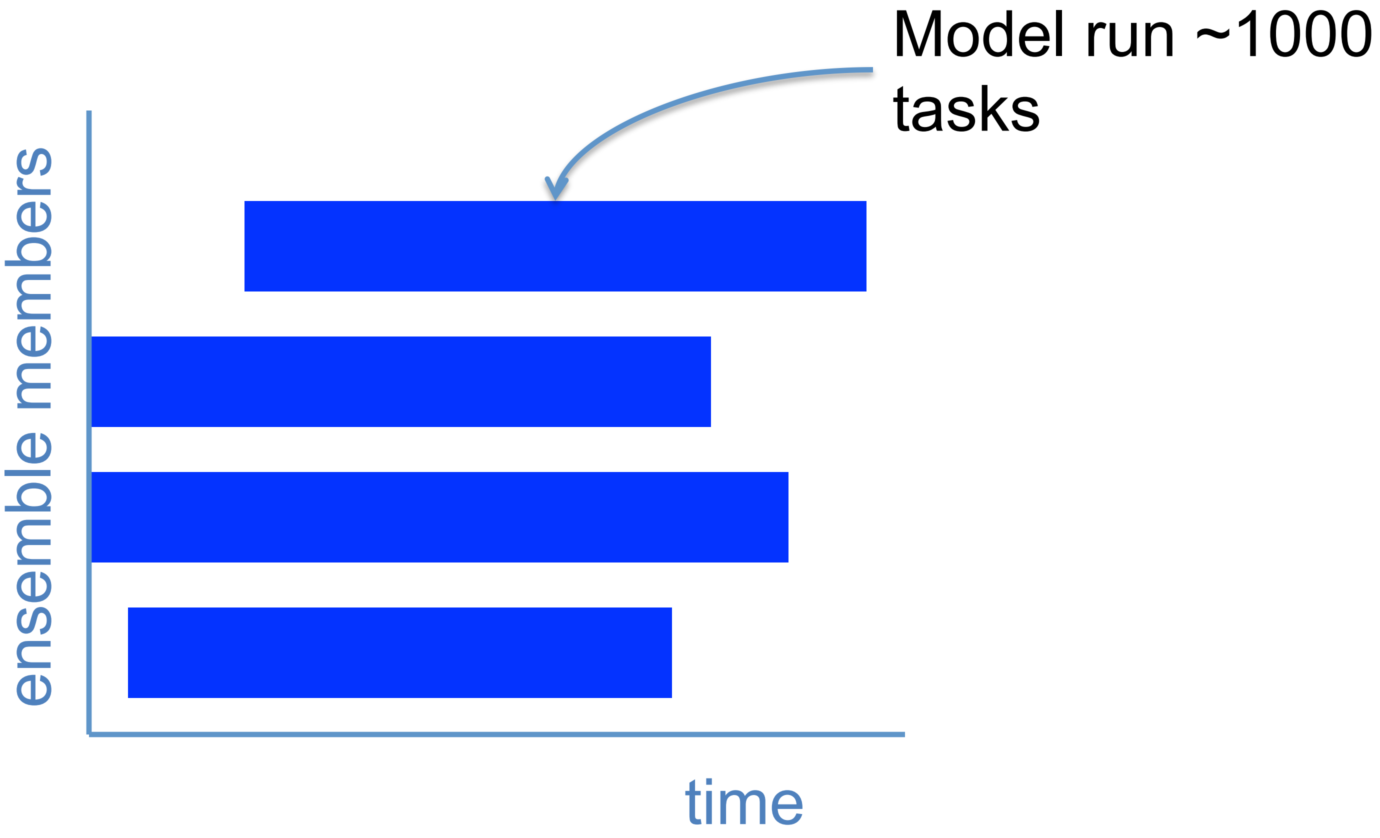
Multiple model forecasts to create the ensemble

IO



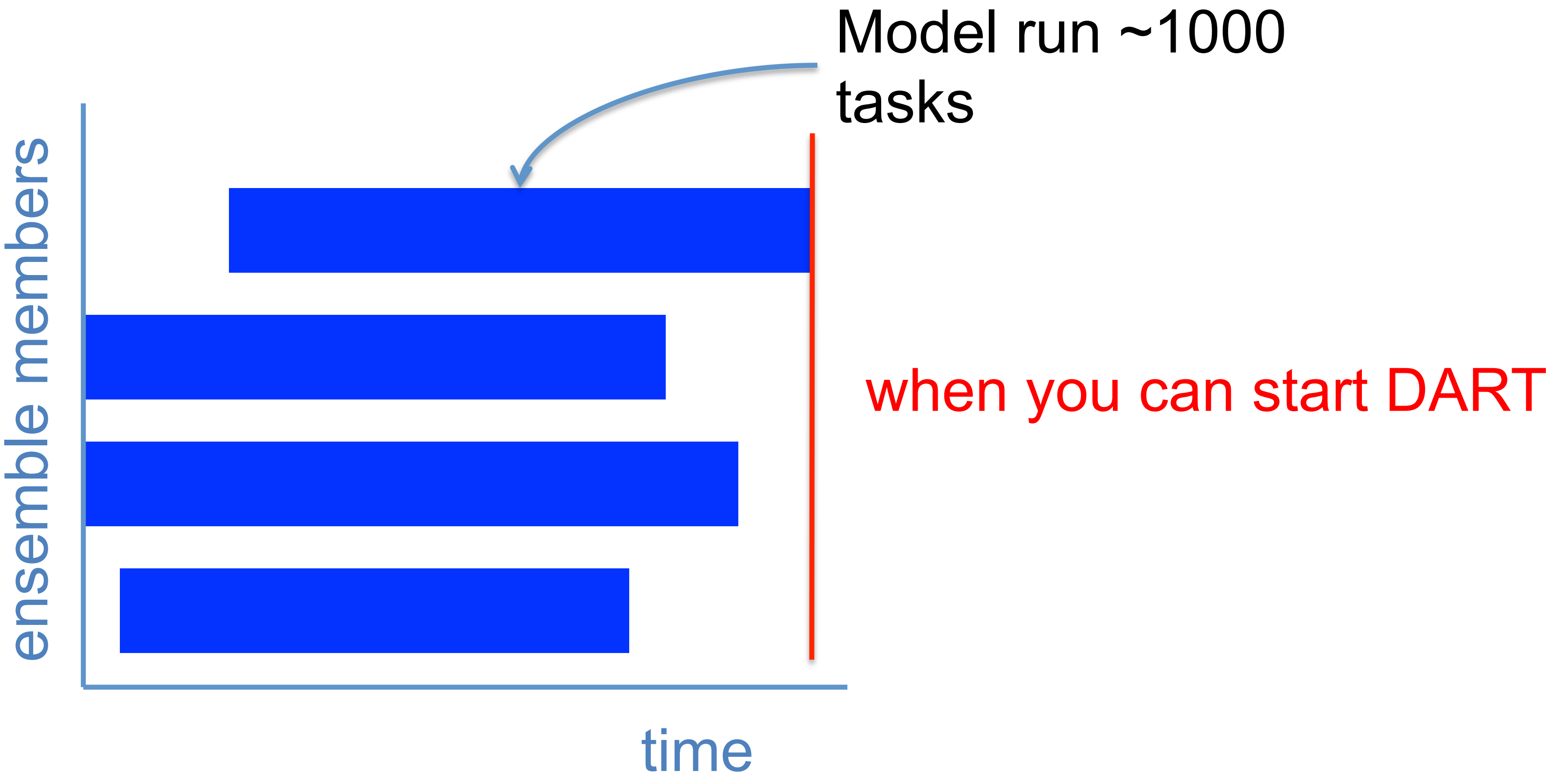
Multiple model forecasts to create the ensemble

IO



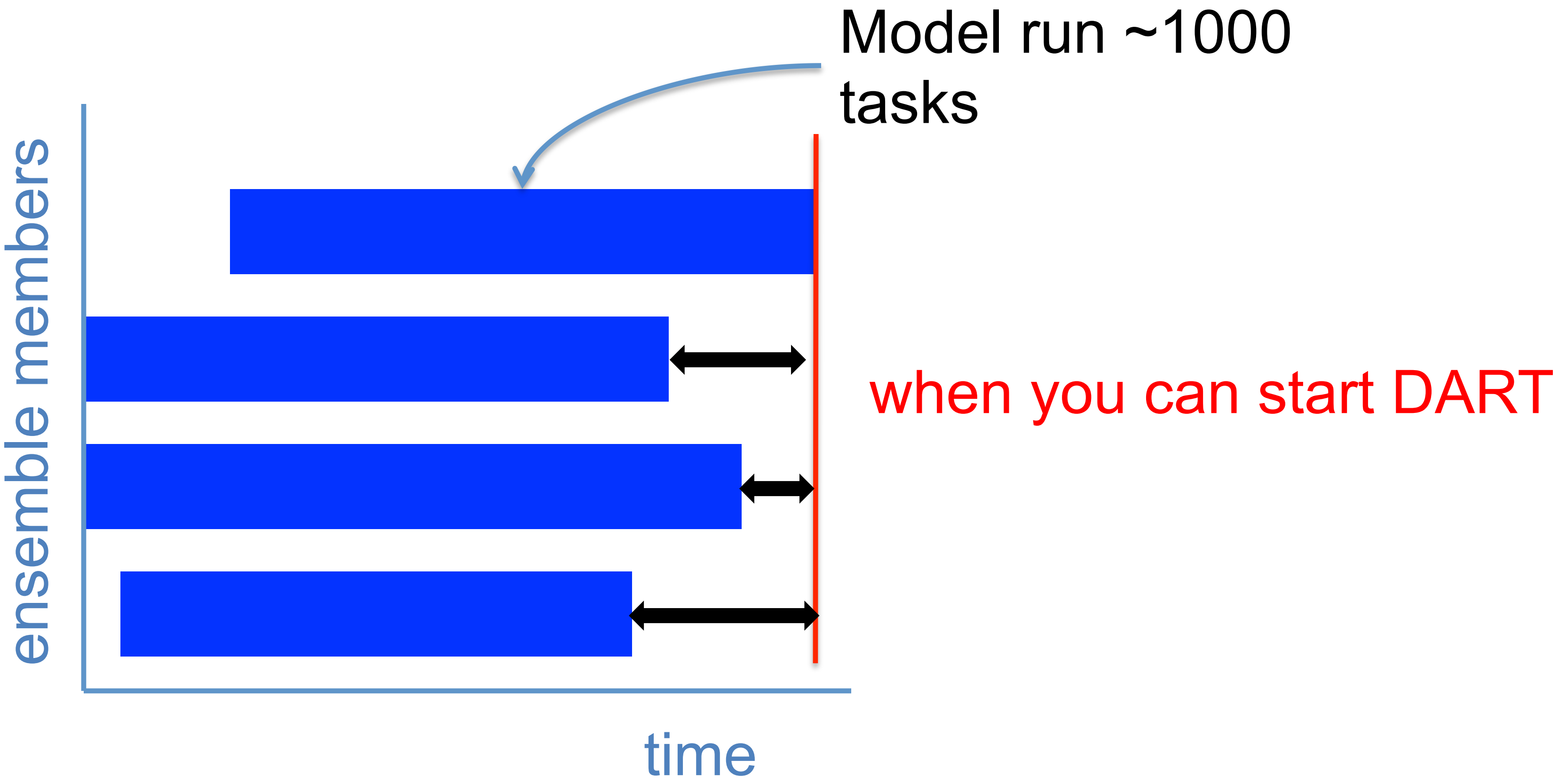
Multiple model forecasts to create the ensemble

IO



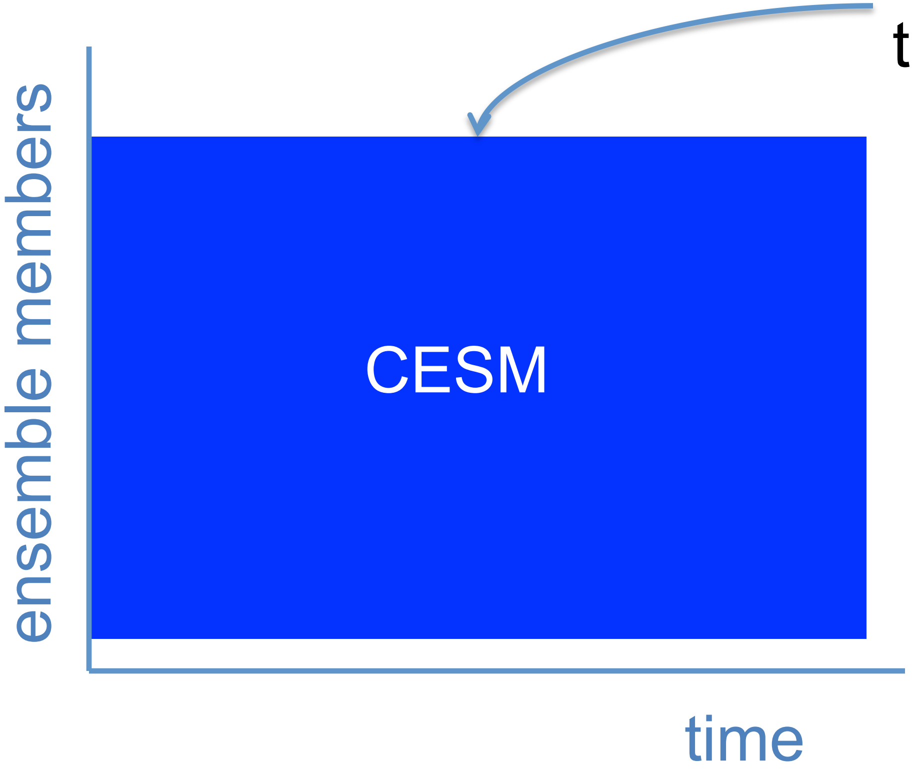
Multiple model forecasts to create the ensemble

IO



Multiple model forecasts to create the ensemble

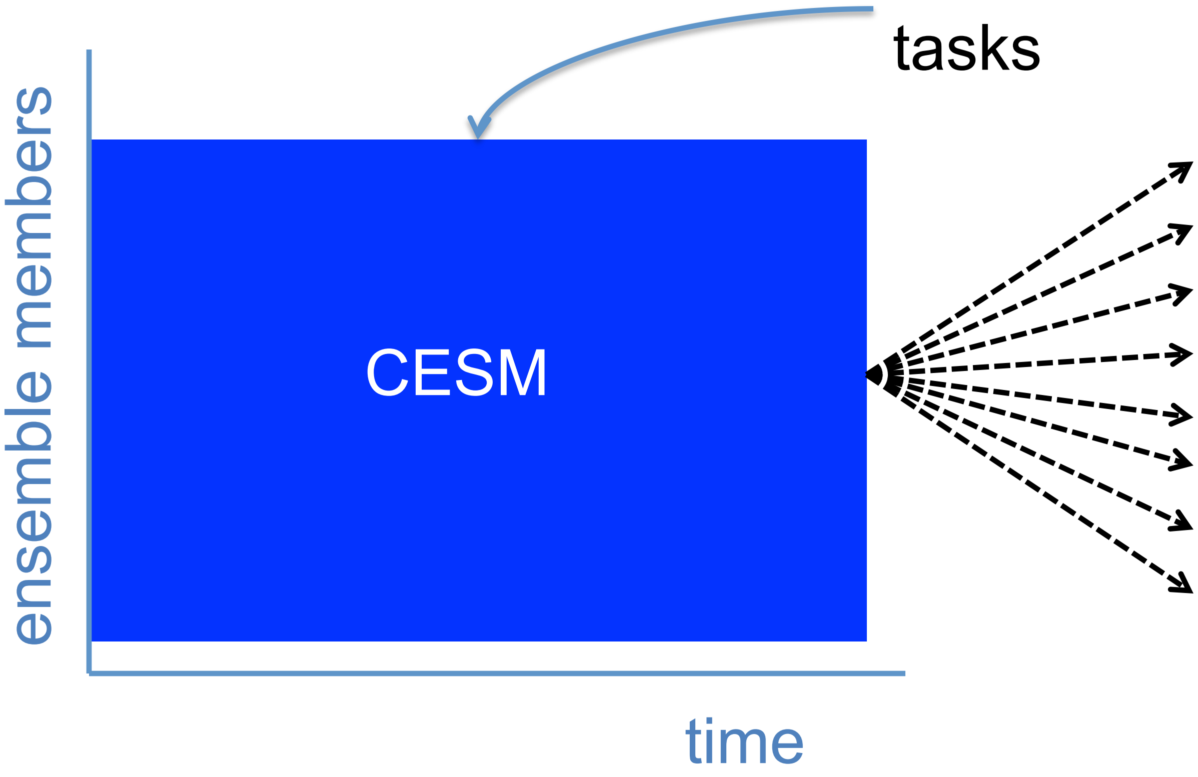
IO



Model run ~10000 tasks

Multi-instance forecasts to create the ensemble

IO

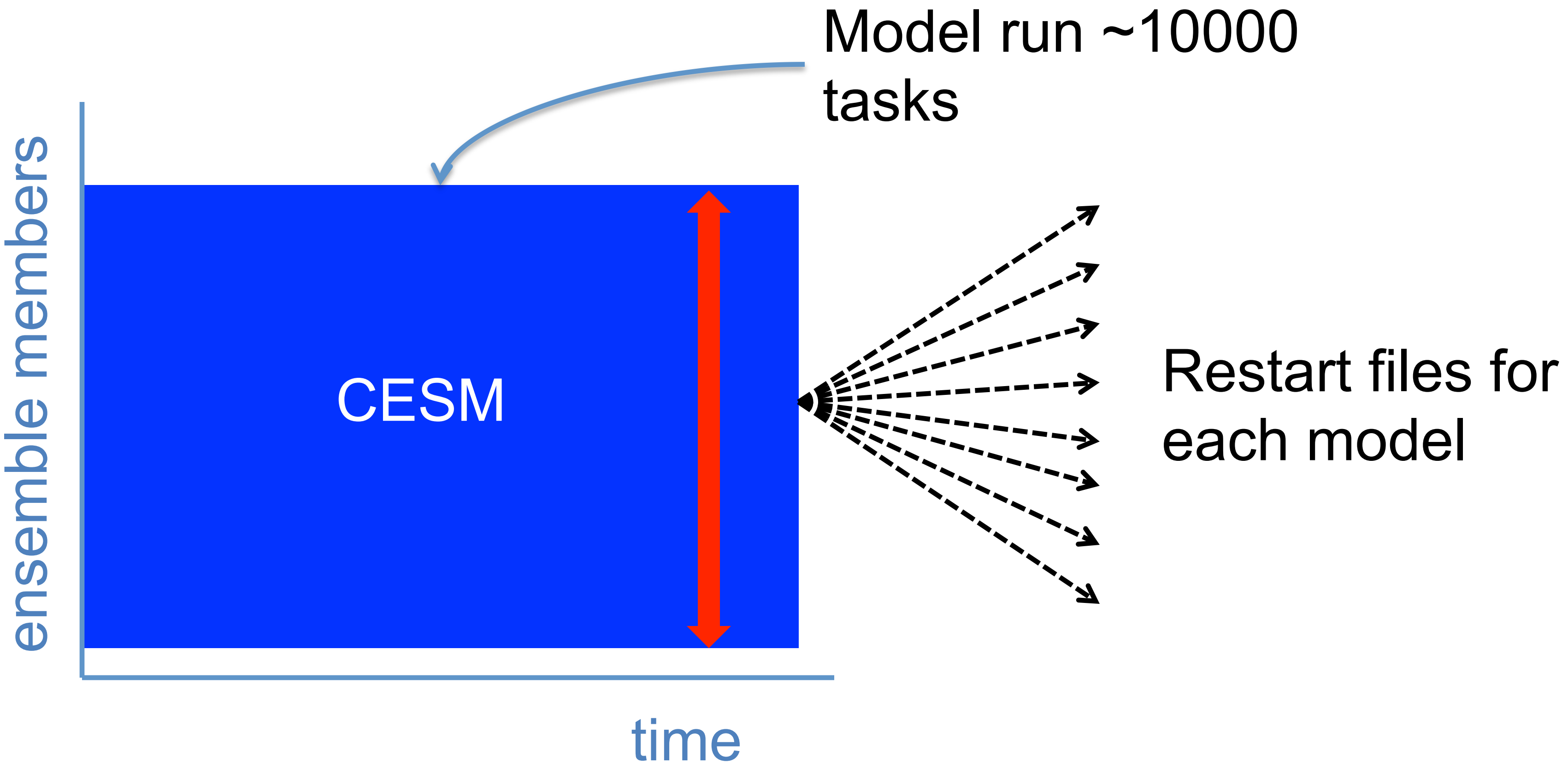


Model run ~10000 tasks

Restart files for each model

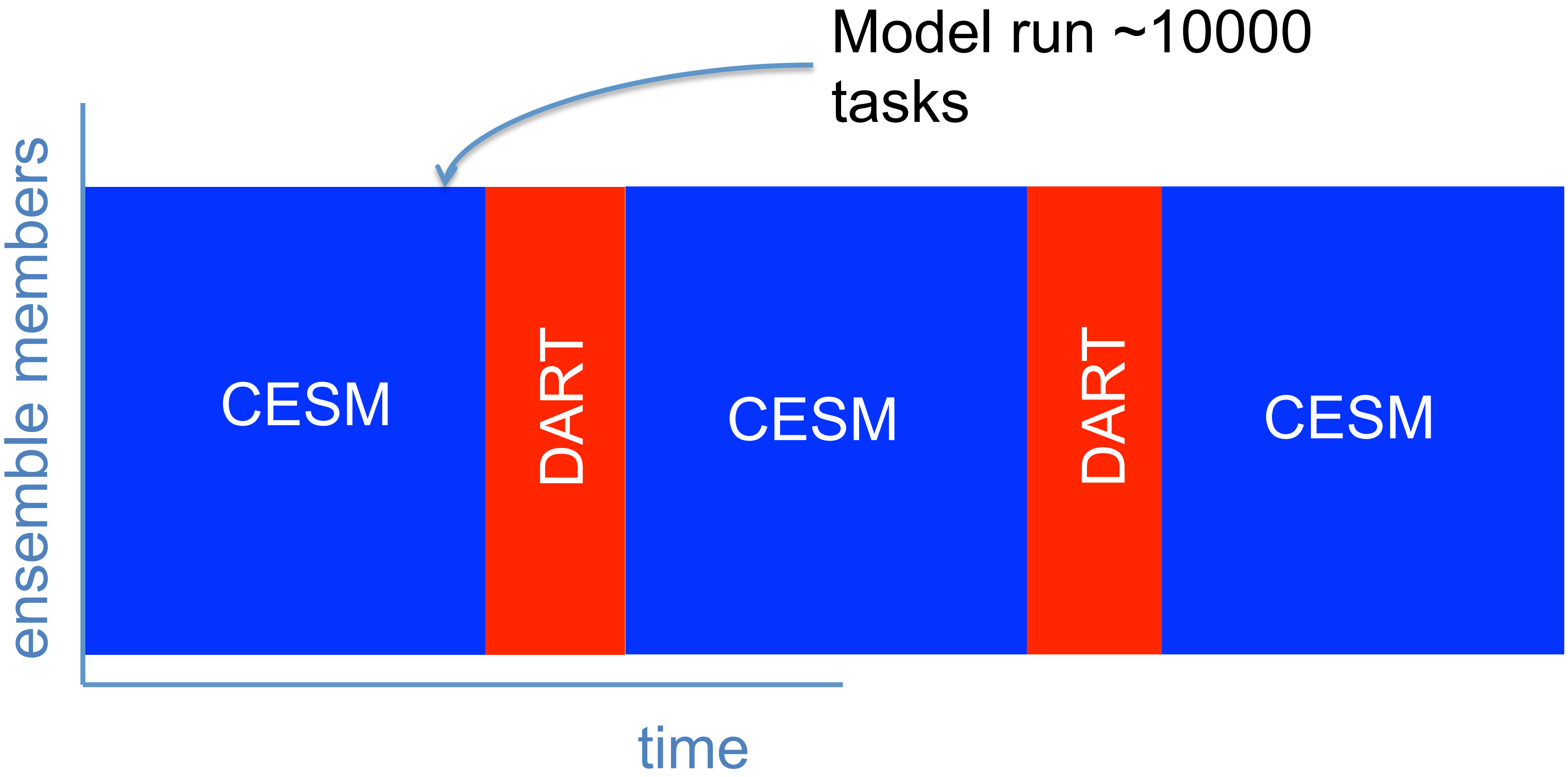
Multi-instance forecasts to create the ensemble

IO



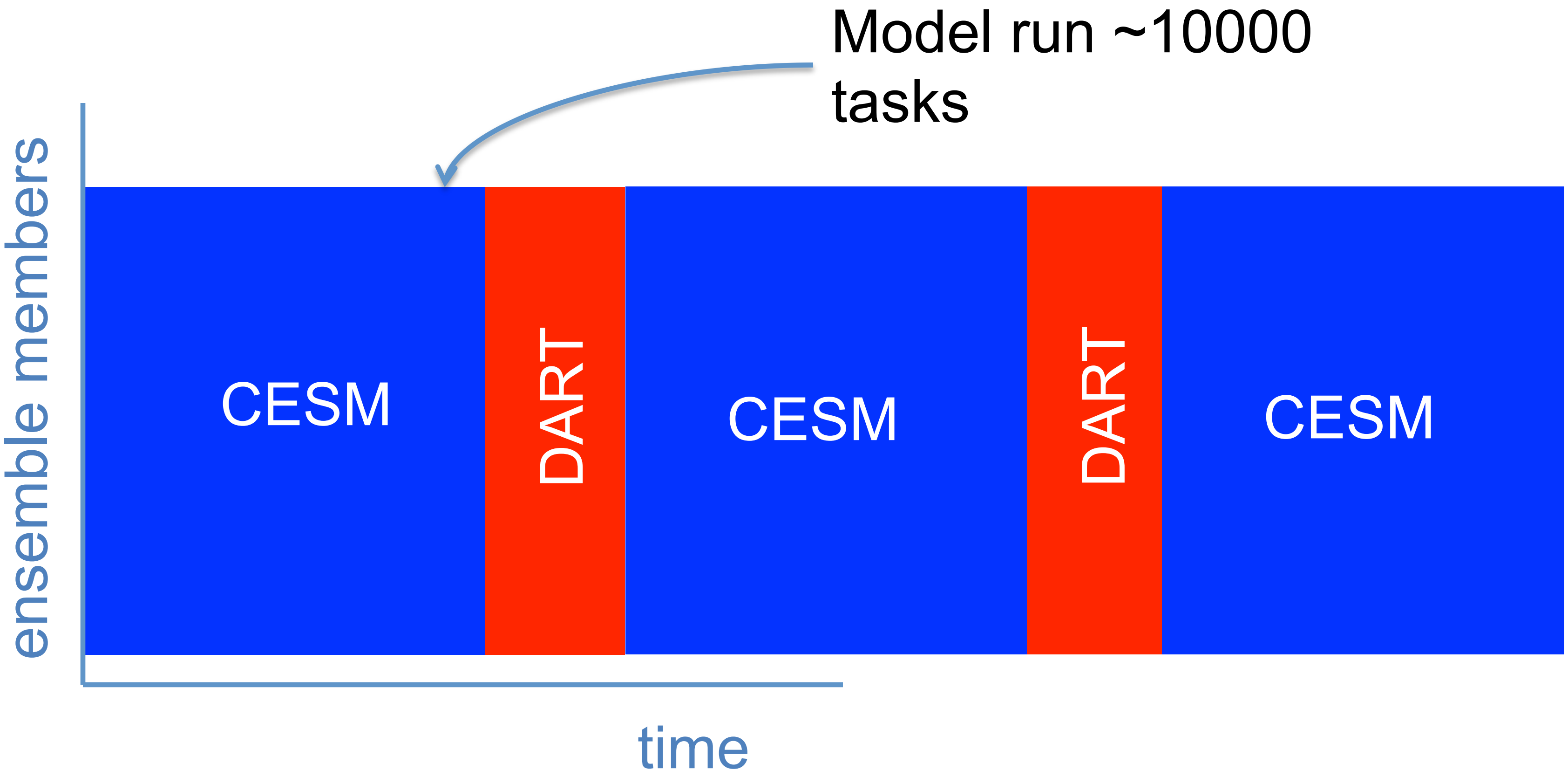
Multi-instance forecasts to create the ensemble

IO



Multi-instance forecasts to create the ensemble

IO



Should the IO speed drive the data layout?

Algorithm choice and communication

- The forward operator parallelizes
- The assimilation parallelizes

Algorithm choice and communication

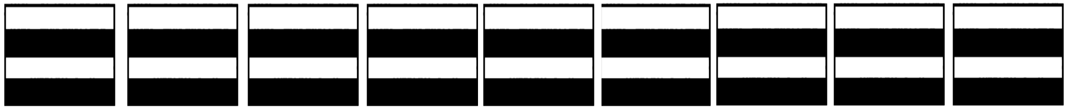
- The forward operator parallelizes
- The assimilation parallelizes
- Communication does not scale

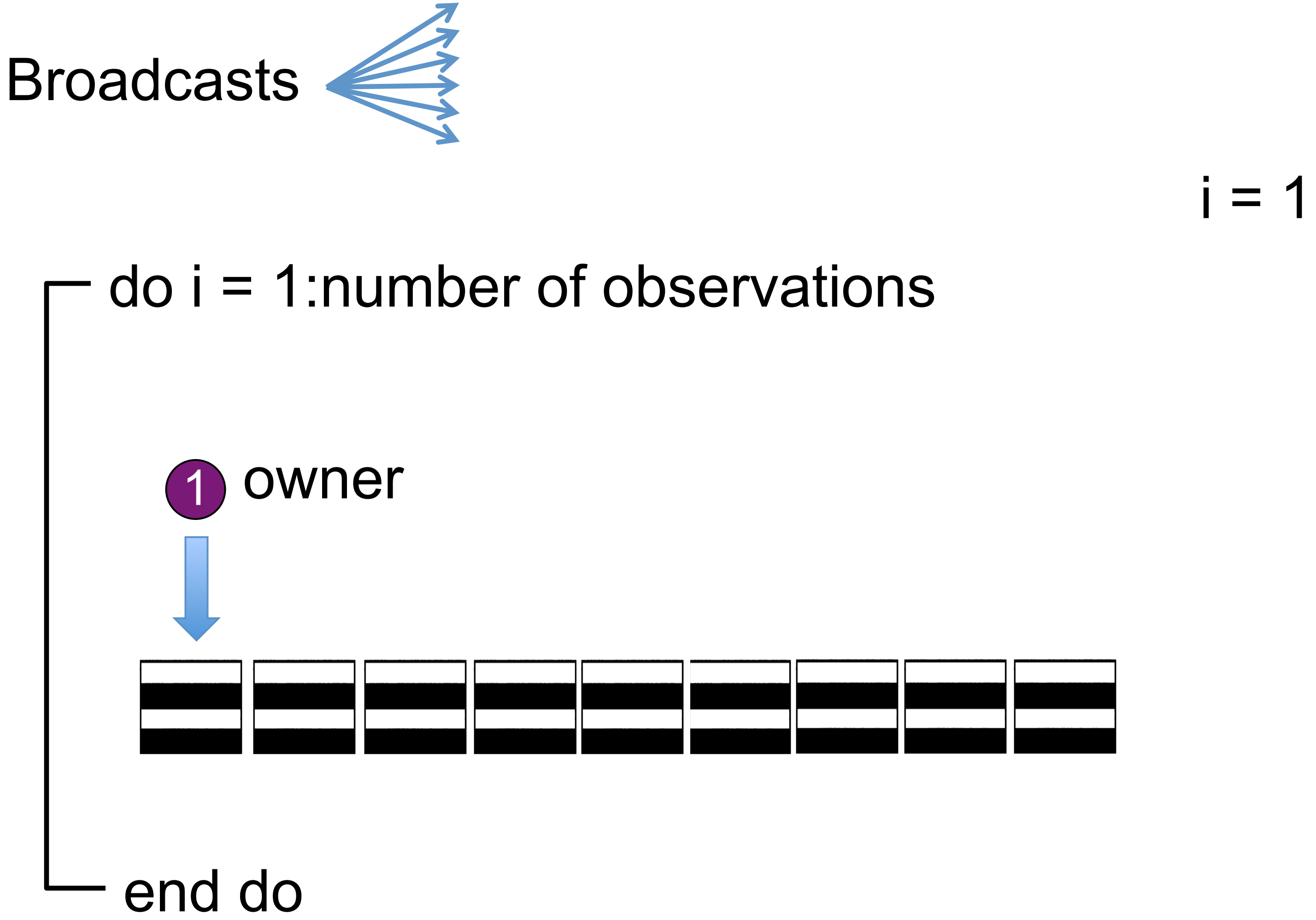
Broadcasts

i = 1

```
do i = 1:number of observations
    ① observation

end do
```





Broadcasts

i = 1

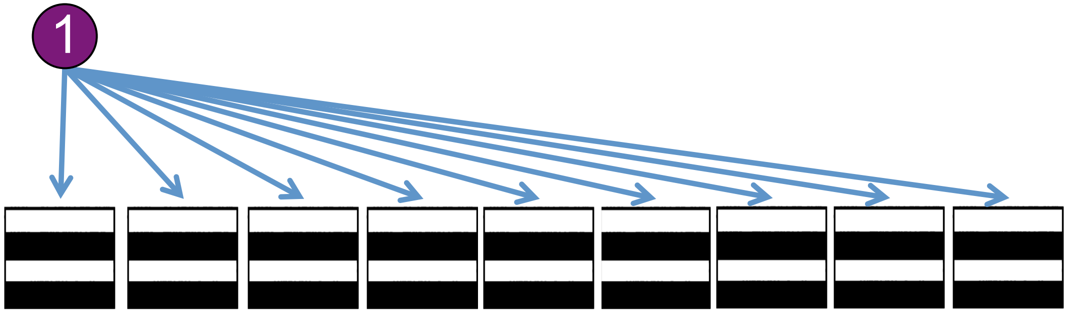
```
do i = 1:number of observations

end do
```

Broadcasts

i = 1

do i = 1:number of observations

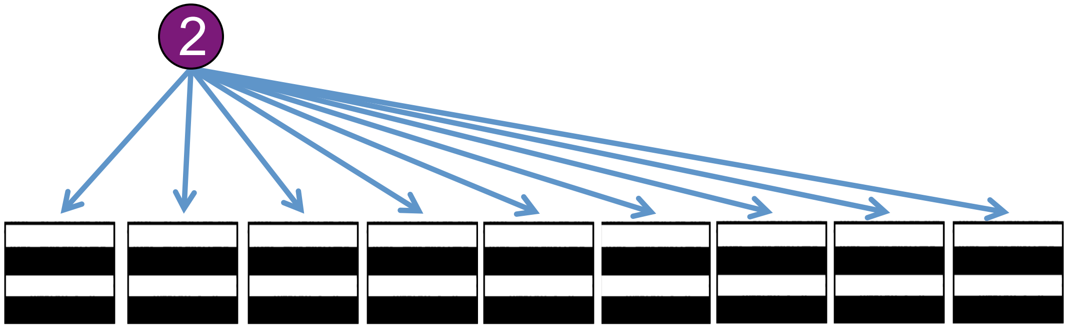


end do

Broadcasts

i = 2

do i = 1:number of observations

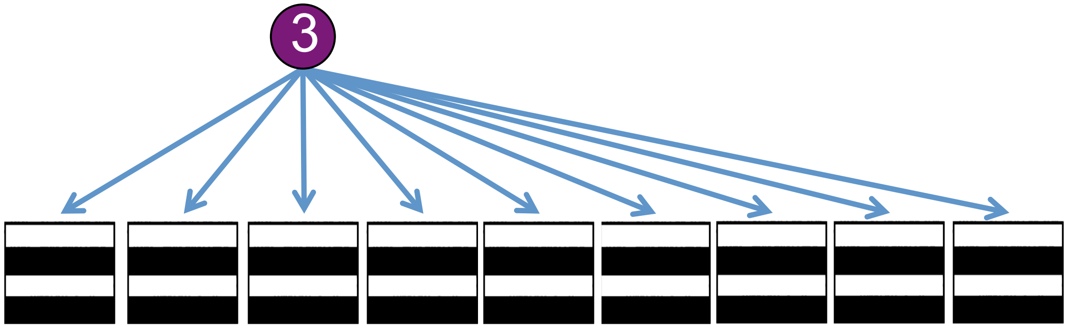


end do

Broadcasts

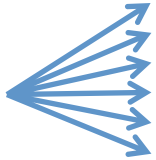
i = 3

do i = 1:number of observations



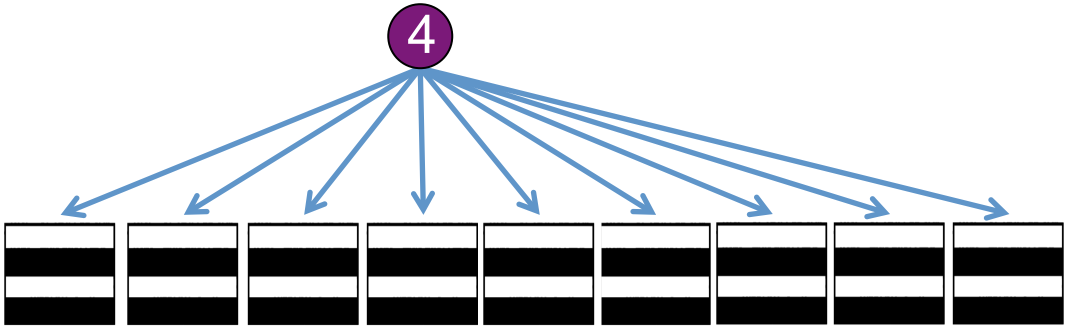
end do

Broadcasts



i = 4

```
do i = 1:number of observations
```

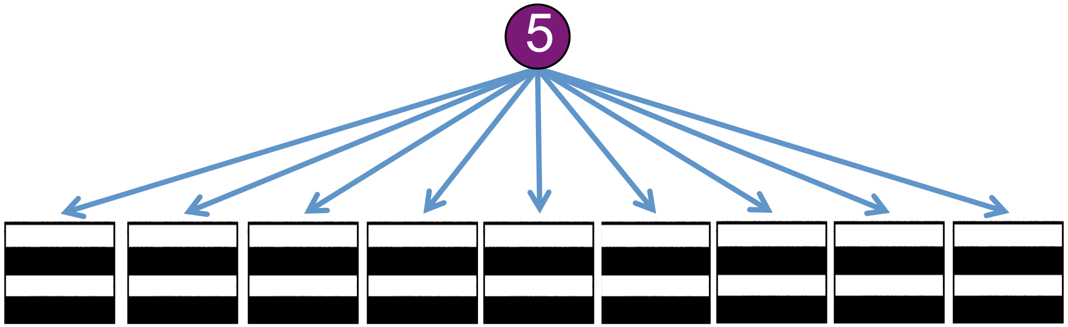


```
end do
```

Broadcasts

i = 5

```
do i = 1:number of observations
```

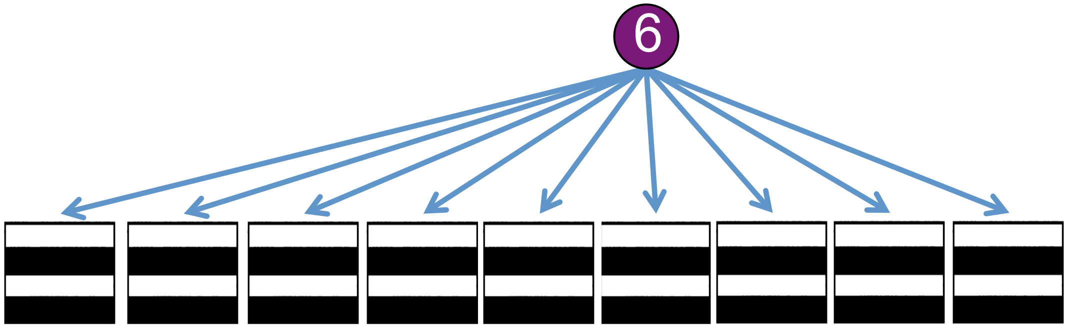


```
end do
```

Broadcasts

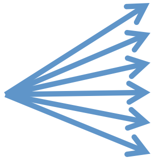
i = 6

do i = 1:number of observations



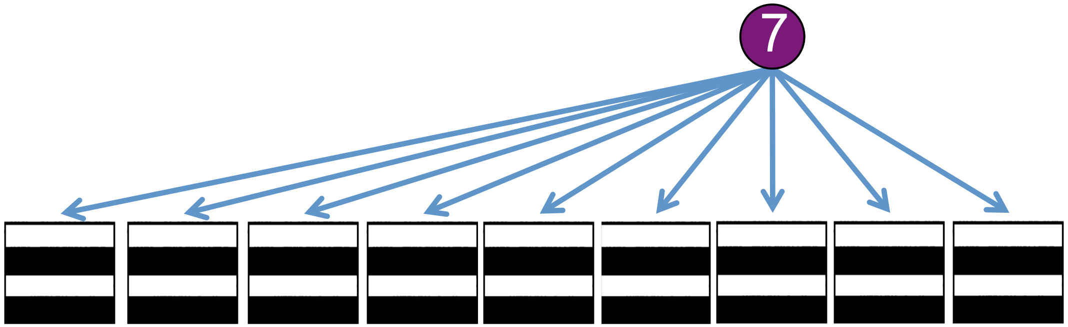
end do

Broadcasts



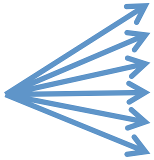
i = 7

do i = 1:number of observations



end do

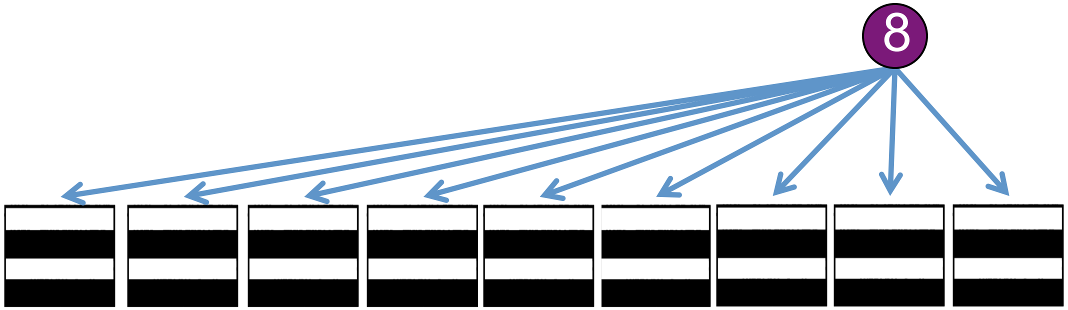
Broadcasts



i = 8

```
do i = 1:number of observations

end do
```



Further Complications

Further Complications

Or, software engineering concerns

Further Complications

Or, software engineering concerns

What about all the users who are happy with DART as it is?

Further Complications

Or, software engineering concerns

What about all the users who are happy with DART as it is?

- Allow whole state to be stored if the memory is available

Further Complications

Or, software engineering concerns

What about all the users who are happy with DART as it is?

- Allow whole state to be stored if the memory is available

Does this mean a vectorized and non-vectorized version of the forward operator for each model?

Further Complications

Or, software engineering concerns

What about all the users who are happy with DART as it is?

- Allow whole state to be stored if the memory is available
- Need to remain user extensible

Further Complications

Or, software engineering concerns

What about all the users who are happy with DART as it is?

- Allow whole state to be stored if the memory is available
- Need to remain user extensible
- Backward compatible?

Further Complications

Or, software engineering concerns

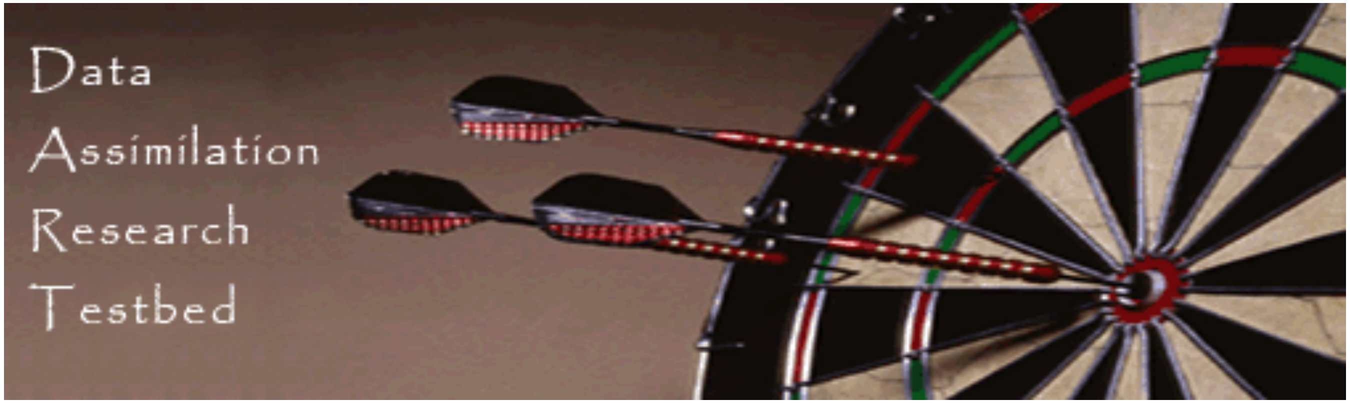
What about all the users who are happy with DART as it is?

- Allow whole state to be stored if the memory is available
- Need to remain user extensible
- Backward compatible?
- Manageable code

Collaborators?

dart@ucar.edu

Learn more about DART at:



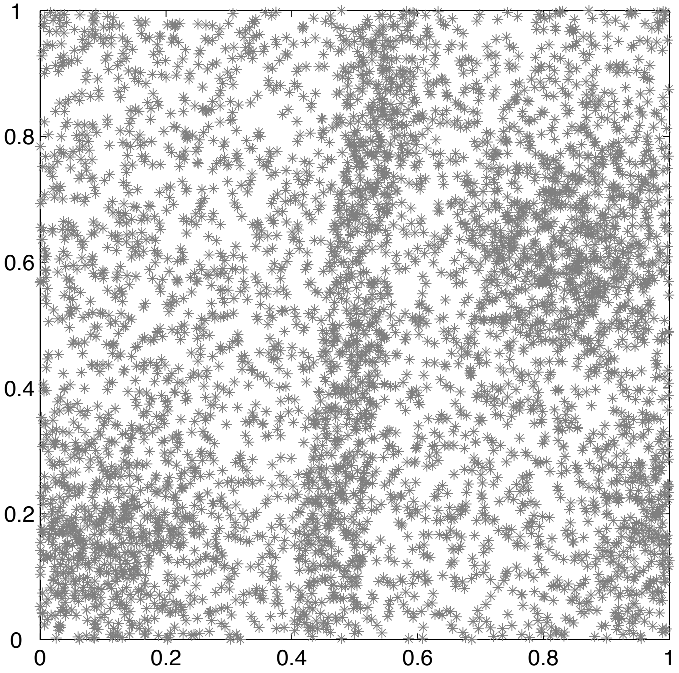
www.image.ucar.edu/DAReS/DART

dart@ucar.edu

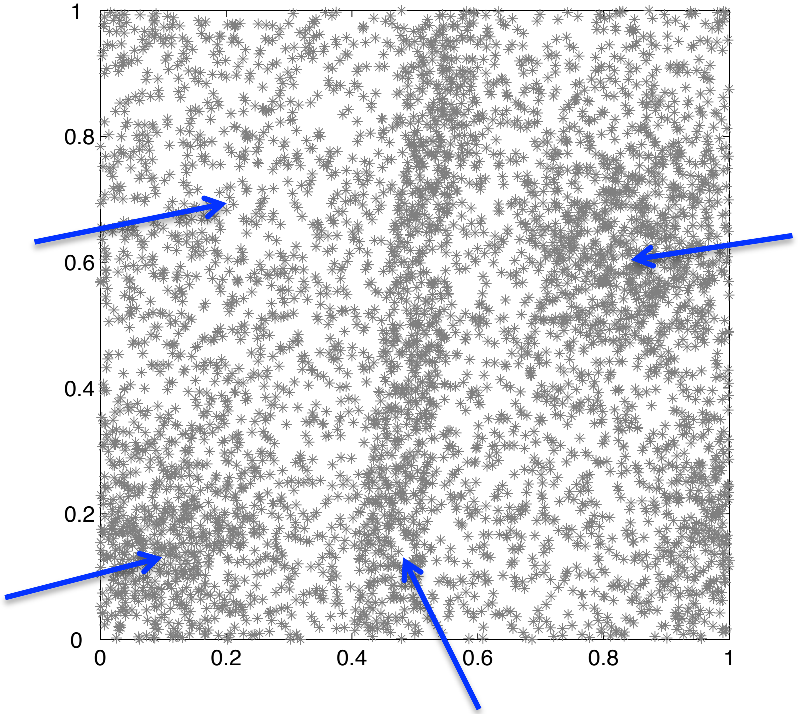
hkershaw@ucar.edu

Parallel Observation Processing

Parallel Observation Processing



Parallel Observation Processing



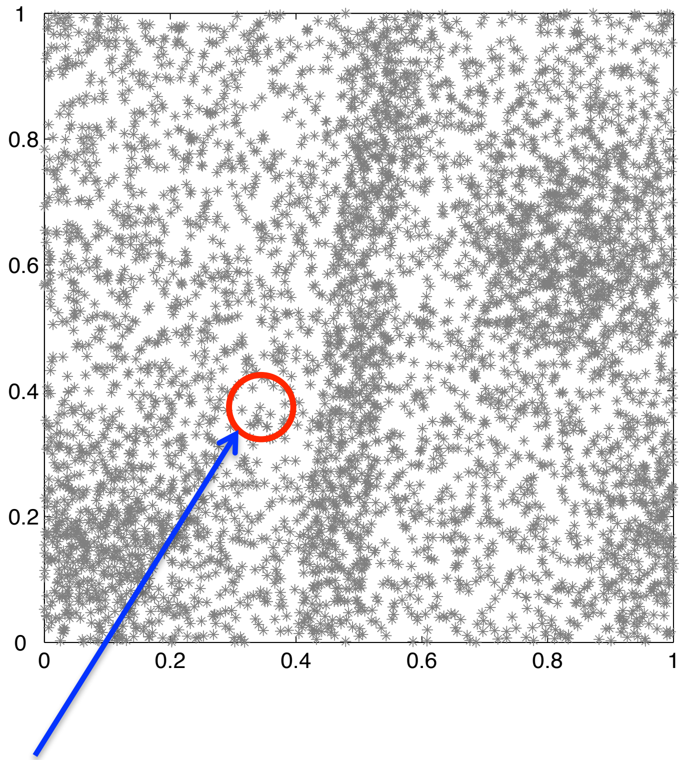
Uniform:
127,000 obs.

Radar:
25,000 obs.

Radar:
25,000 obs.

Satellite track:
25,000 obs.

Parallel Observation Processing



Observations that are more than 0.05 apart are independent.

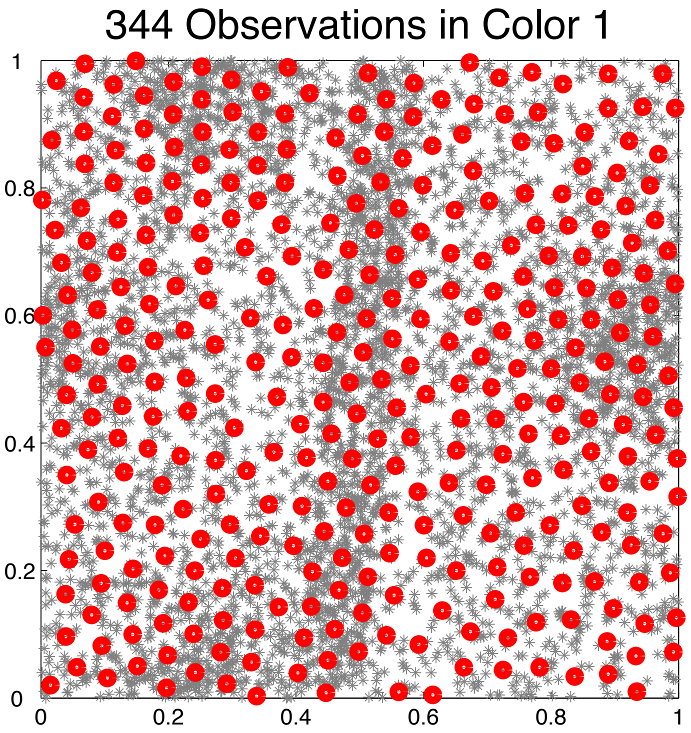
Parallel Observation Processing

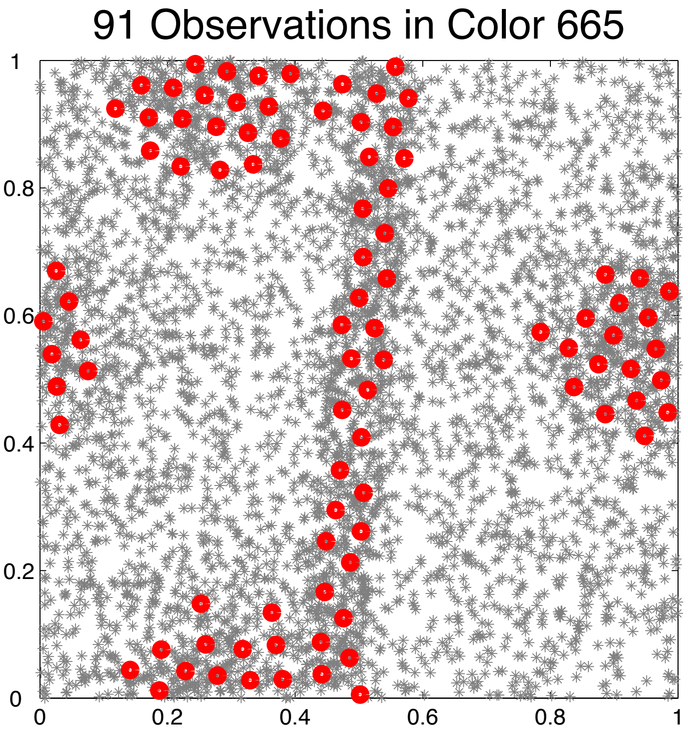
- Find minimum number of subsets of independent observations
- Mutual exclusion scheduling problem
- Use greedy algorithm:

  Decreasing Greedy Mutual Exclusion (DGME)

Parallel Observation Processing

Red shows observations in a given subset.





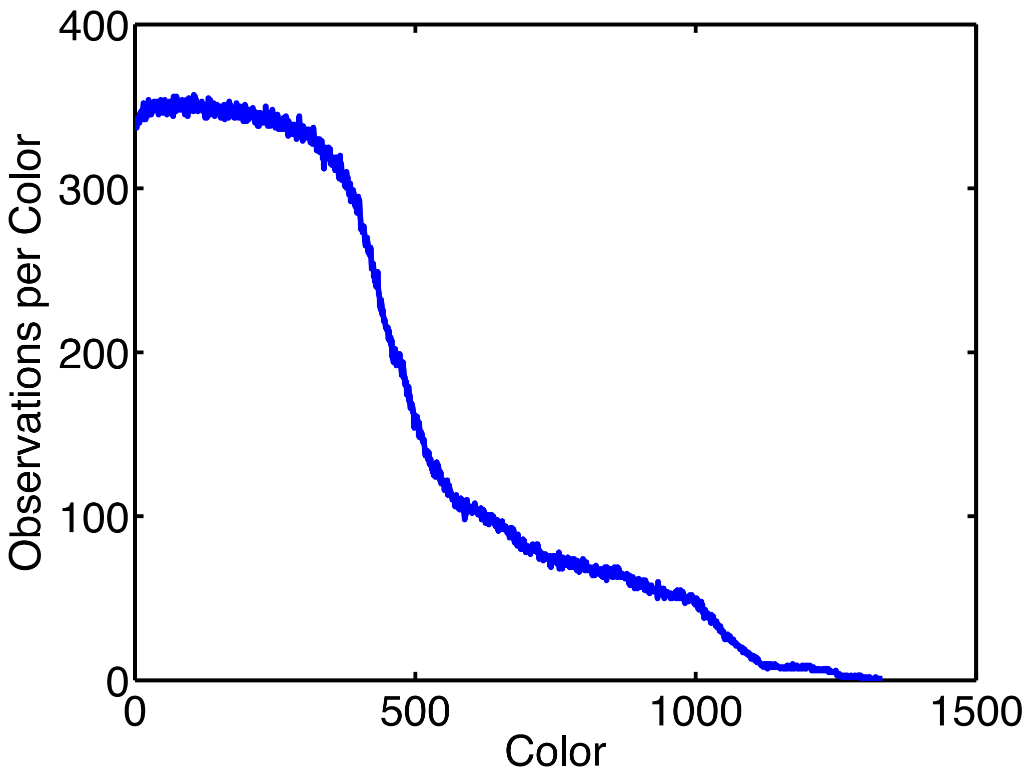
Irregular Observations -> Load Balance Challenges

Parallel Observation Processing

Last subsets only have a few observations each.

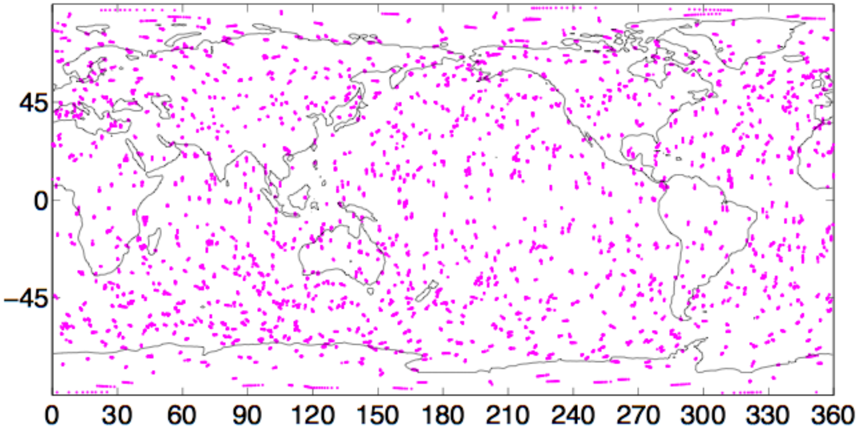
-These are in regions where satellite and radar overlapped.

- May be significant load balance issue.

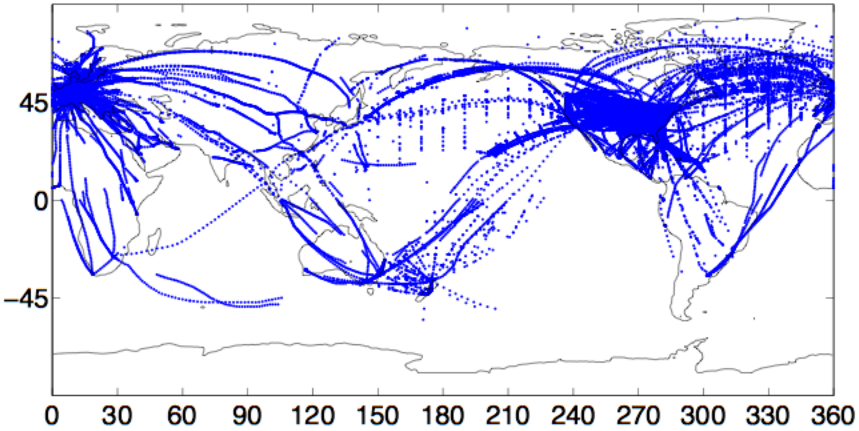


Observations 1 December 2006

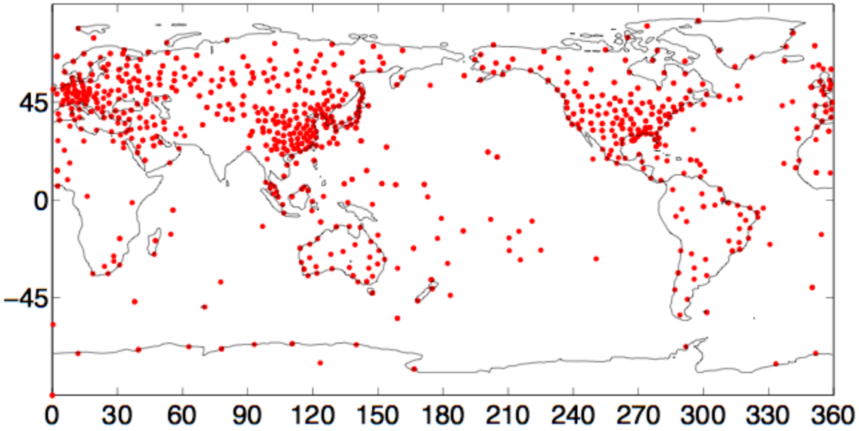
GPS



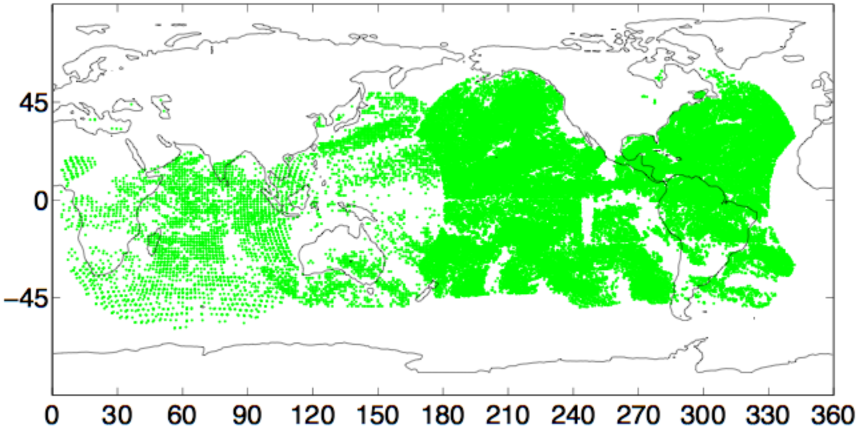
ACARS and Aircraft



Radiosondes



Sat Winds



Parallel netcdf

- Can we use this to transpose during IO?

- Simple for DART restart files

- Not simple for model restart files

Parallel netcdf

- Can we use this to transpose during IO?
- Simple for DART restart files
  - stride through a vector
- Not simple for model restart files

Parallel netcdf

- Can we use this to transpose during IO?

- Simple for DART restart files
  - stride through a vector

- Not simple for model restart files
  - can’t ignore the dimensionality of each variable

Parallel netcdf

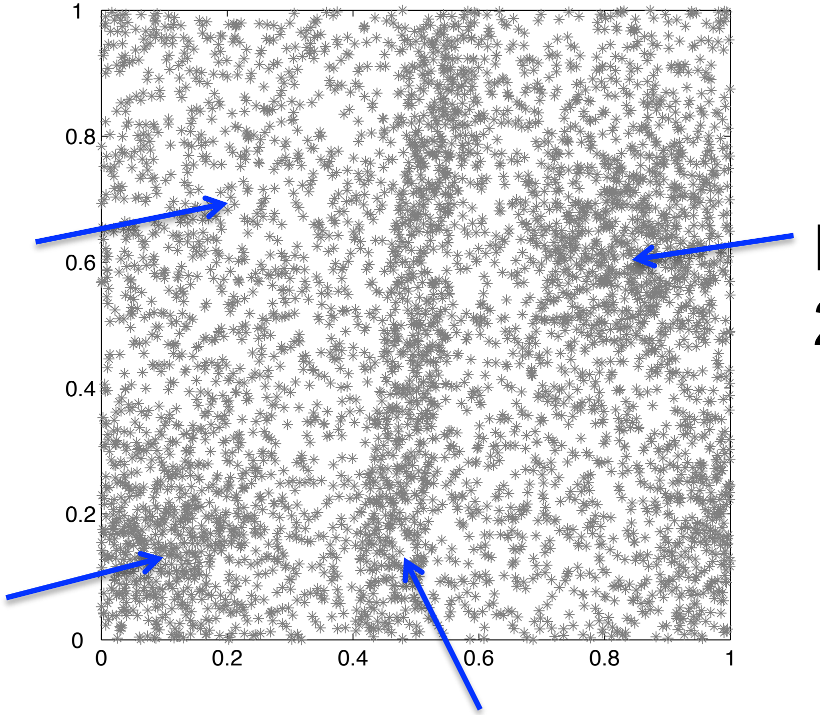
- Can we use this to transpose during IO?
- Simple for DART restart files
  - stride through a vector
- Not simple for model restart files
  - can’t ignore the dimensionality of each variable

Parallel netcdf

- Can we use this to transpose during IO?
- Simple for DART restart files
  - stride through a vector
- Not simple for model restart files
  - can’t ignore the dimensionality of each variable
- Should the IO speed drive the assimilation data layout?

Irregular Observations -> Load Balance Challenges

Simulate performance for idealized observation set (2% of obs shown).



Uniform:
127,000 obs.

Radar:
25,000 obs.

Radar:
25,000 obs.

Satellite track:
25,000 obs.

IO

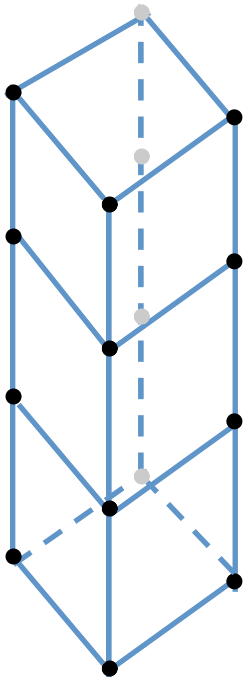
You need to run a bunch of model forecasts

Convert the model output to DART format

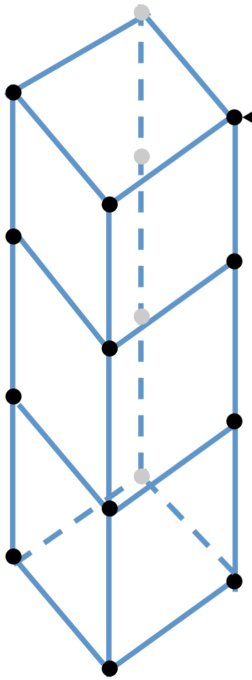
Do data assimilation with DART

Convert back to model input

Calculation of the Forward Operator

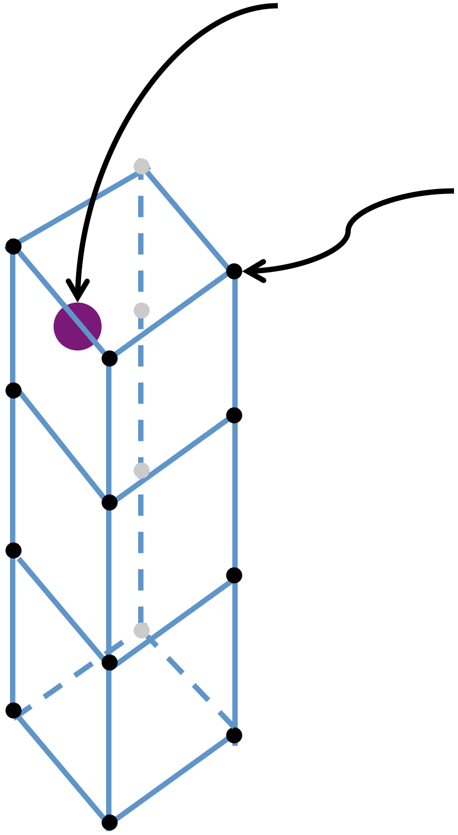


Calculation of the Forward Operator



grid points at each model level

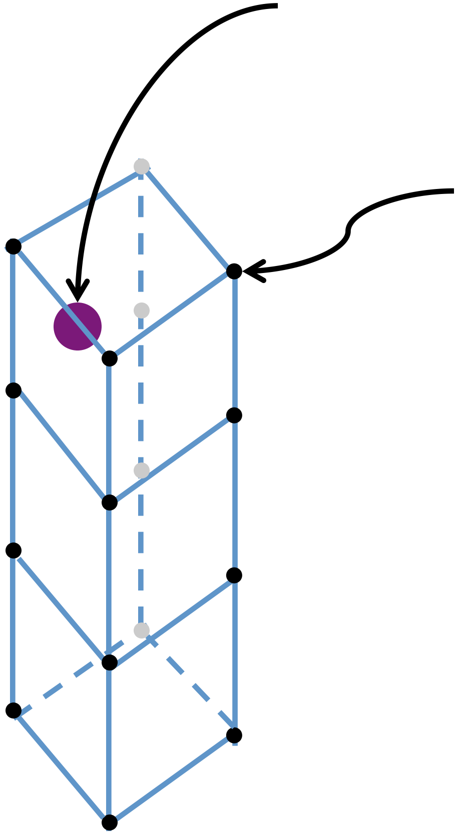
Calculation of the Forward Operator



Observation – at a vertical location
in pressure/height/…

grid points at each model level

Calculation of the Forward Operator

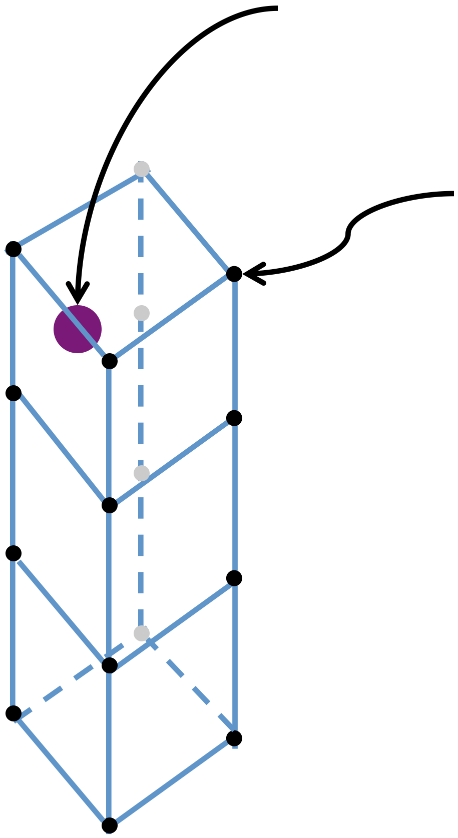


Observation – at a vertical location
in pressure/height/…

grid points at each model level

The variables in the state determine the location of the observation

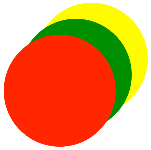
Calculation of the Forward Operator



Observation – at a vertical location
in pressure/height/…

grid points at each model level

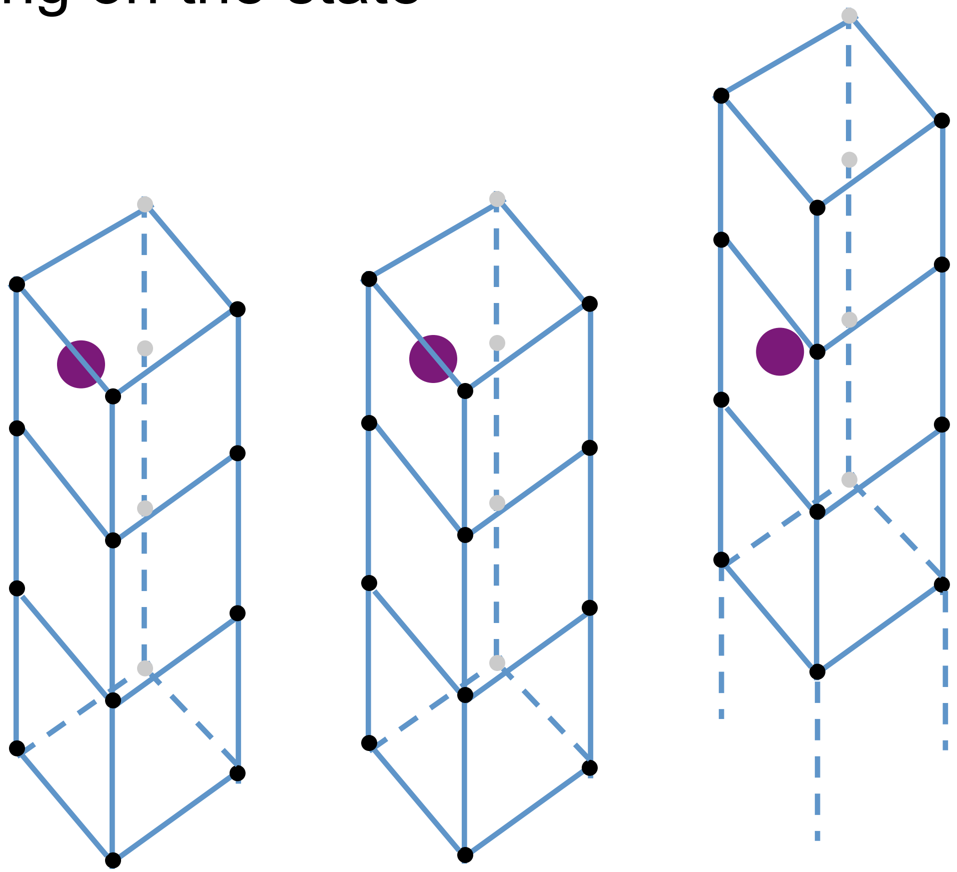
The variables in the state determine the location of the observation



Interpolate to find the expected value of the observation

But vectorization is not perfect:

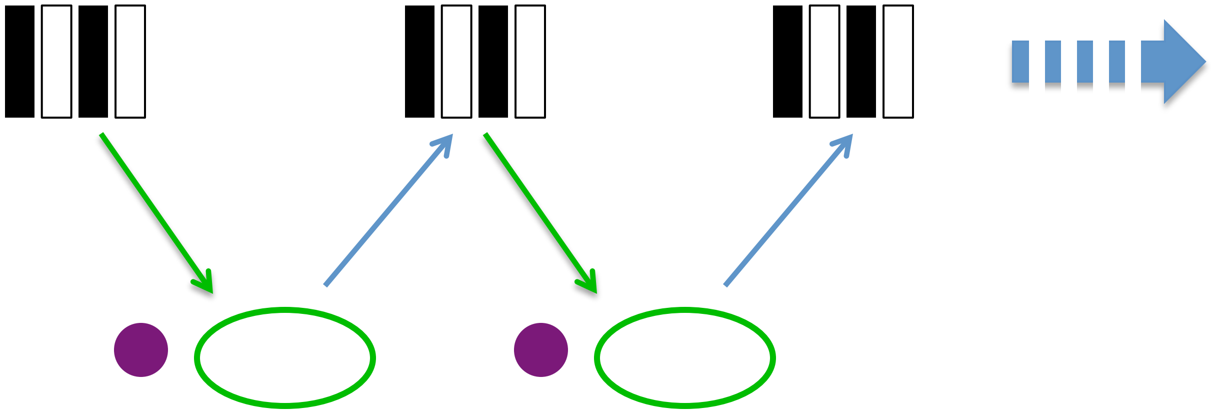
An observation can be in different model levels depending on the state



● Observation

What’s parallel about DART?

First, look at the serial version of the algorithm

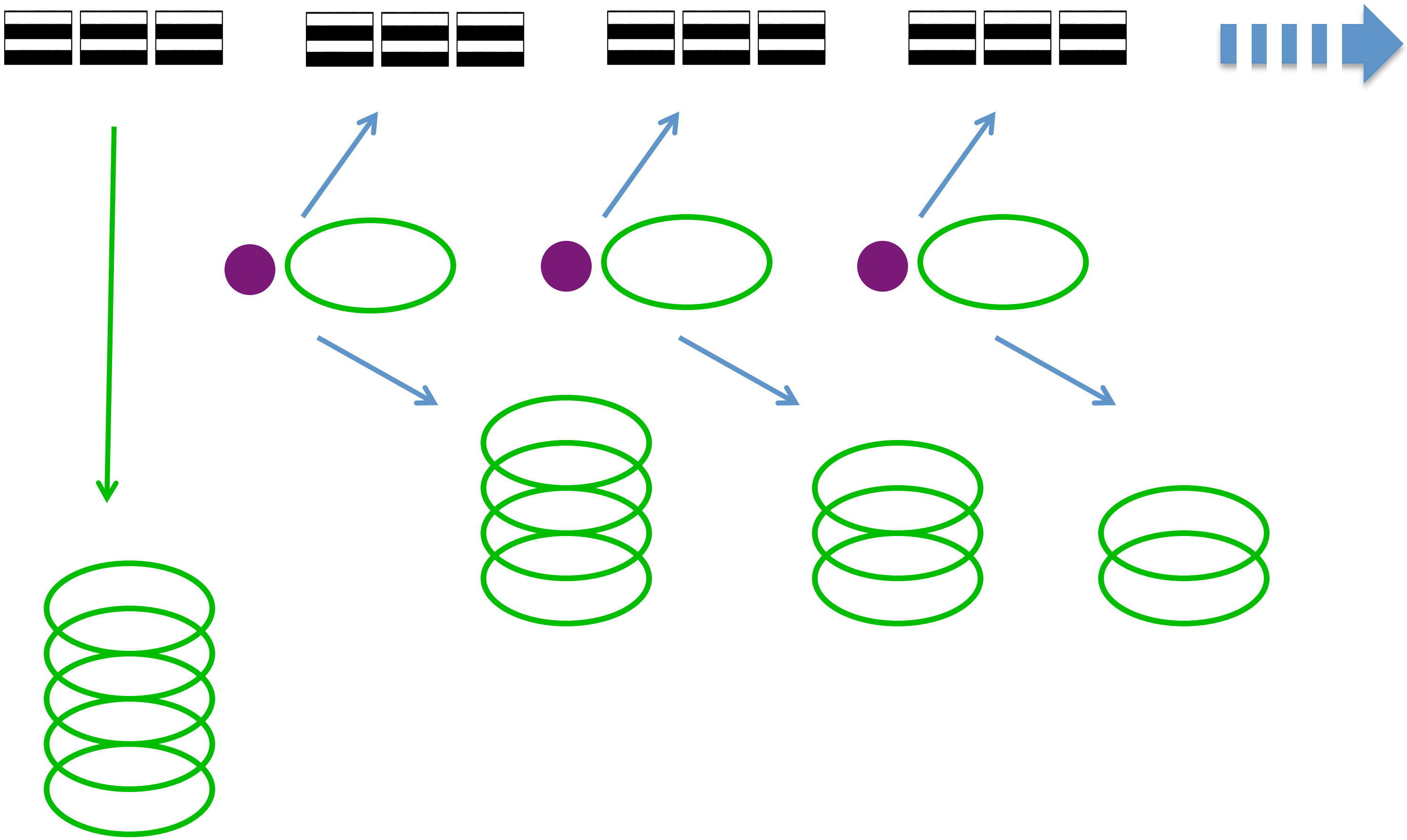


observation and error variance ●

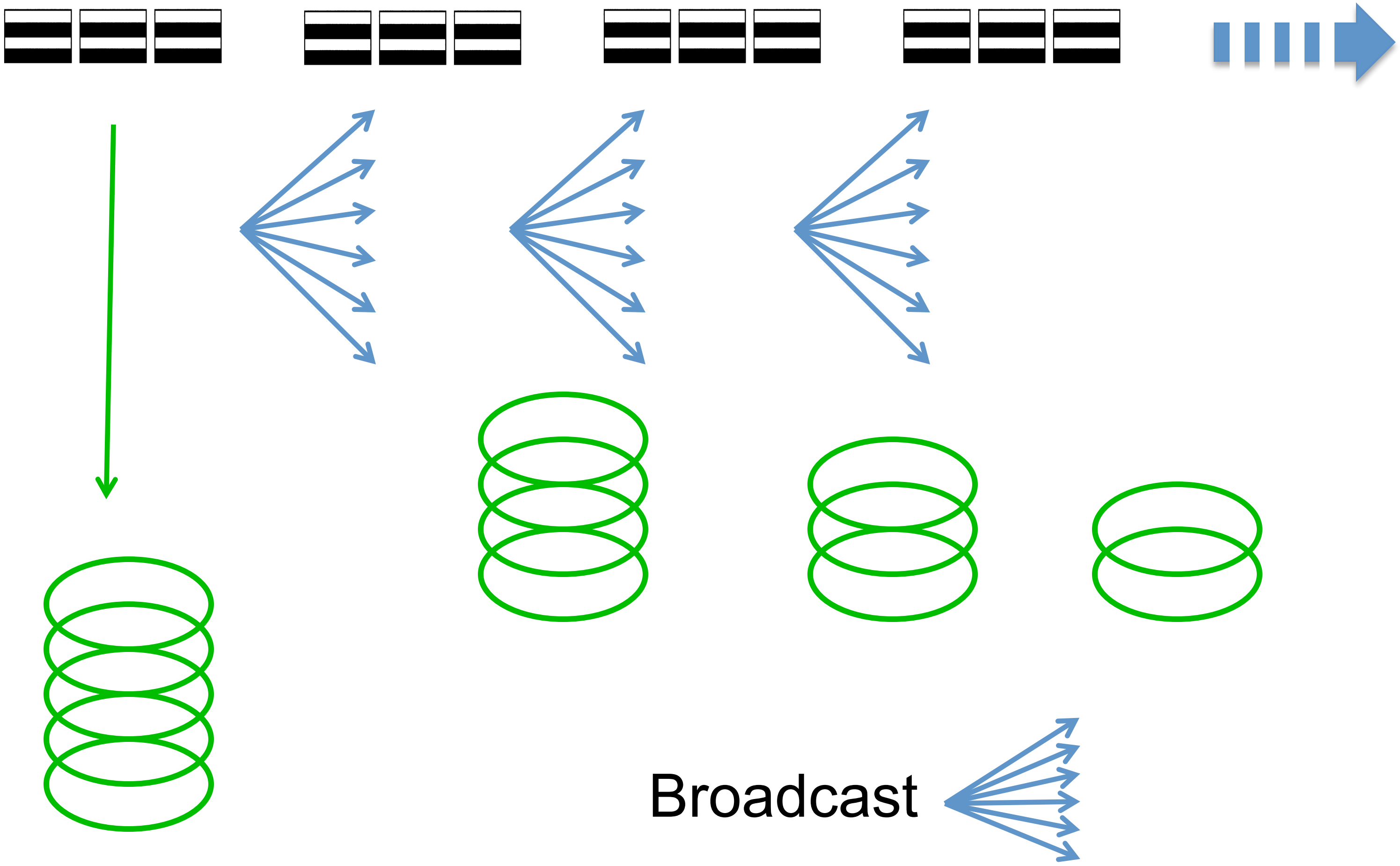
ensemble approximation of the observation ◯

updates →

Algorithm choice and communication



Algorithm choice and communication



IO

Worst-case scenario

IO

You need to run a bunch of model forecasts write to file

IO

You need to run a bunch of model forecasts	write to file
Convert the model output to DART format	read from file write to file

IO

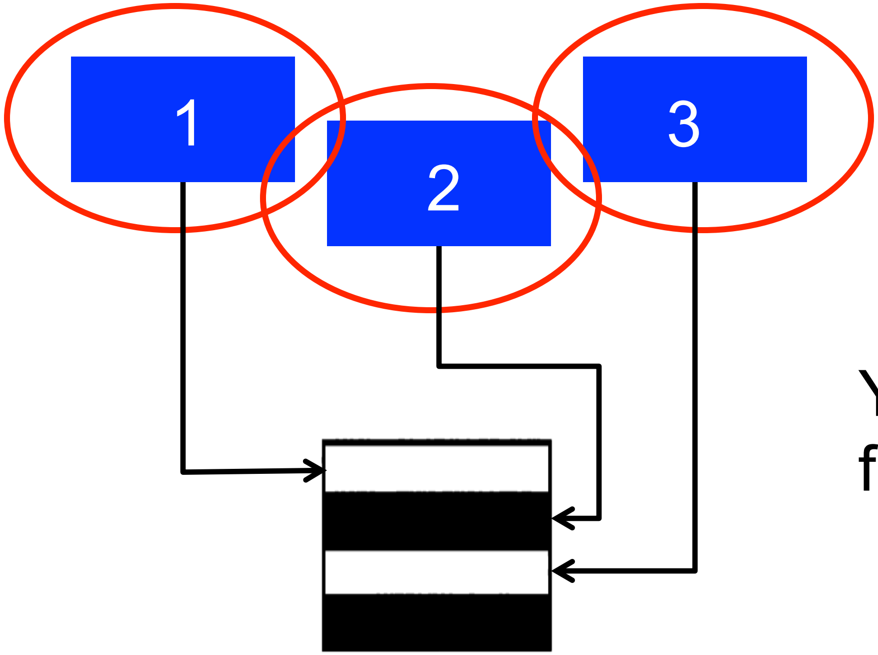
You need to run a bunch of model forecasts	write to file
Convert the model output to DART format	read from file write to file
Do data assimilation with DART	read from file write to file

IO

You need to run a bunch of model forecasts	write to file
Convert the model output to DART format	read from file write to file
Do data assimilation with DART	read from file write to file
Convert back to model input	read from file write to file

IO

Models do not run ensemble complete

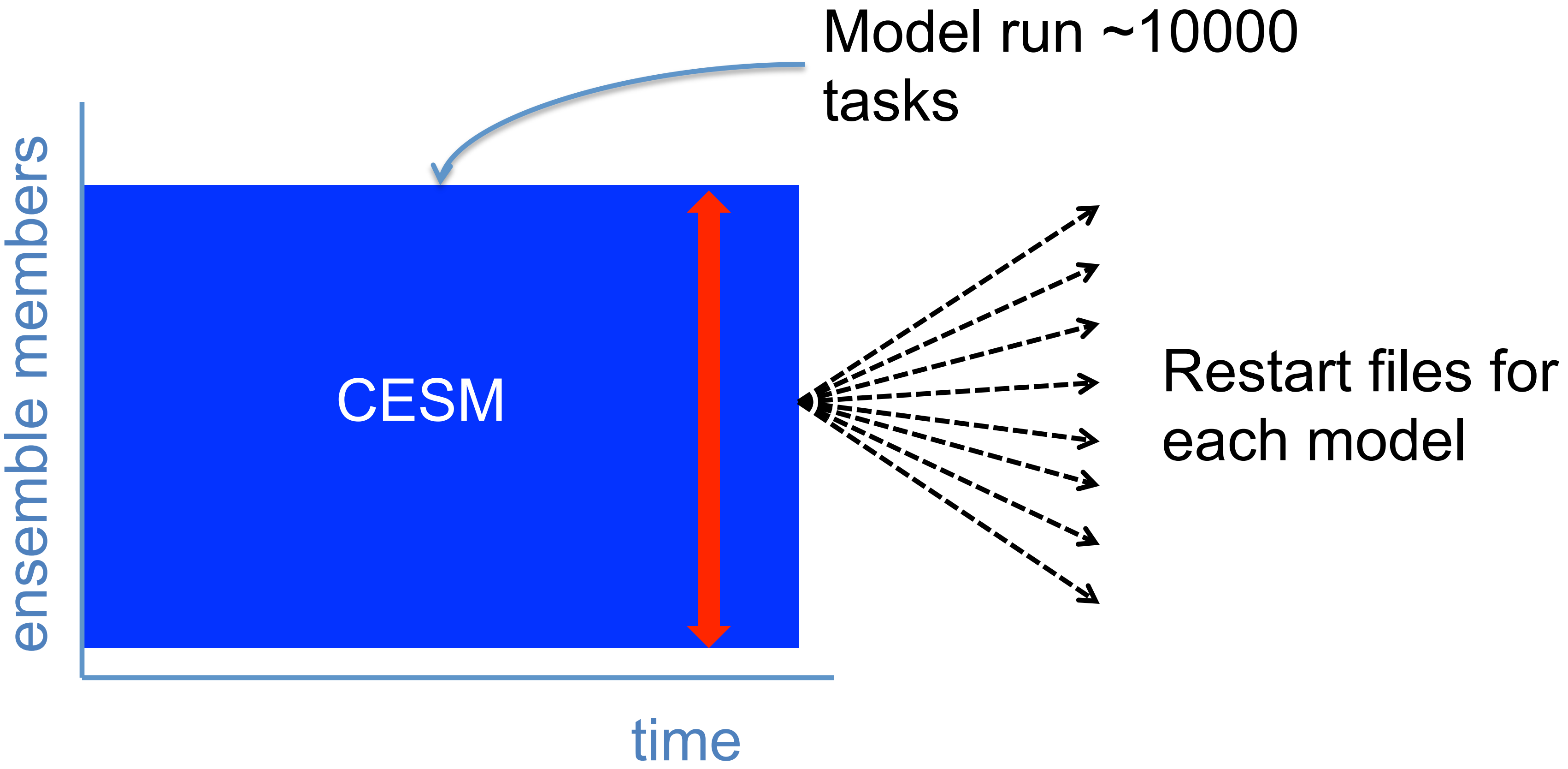


You have to move data from the model to DART

IO

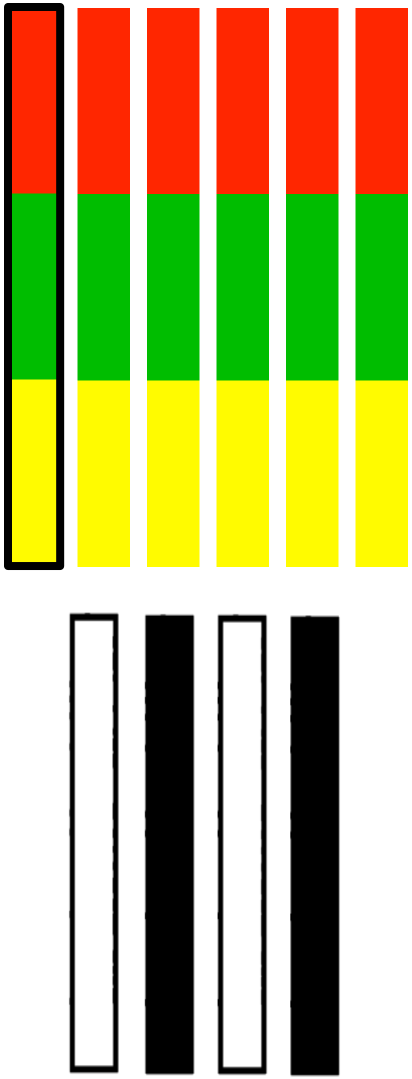
- Scripting
- Queuing
- Scaling

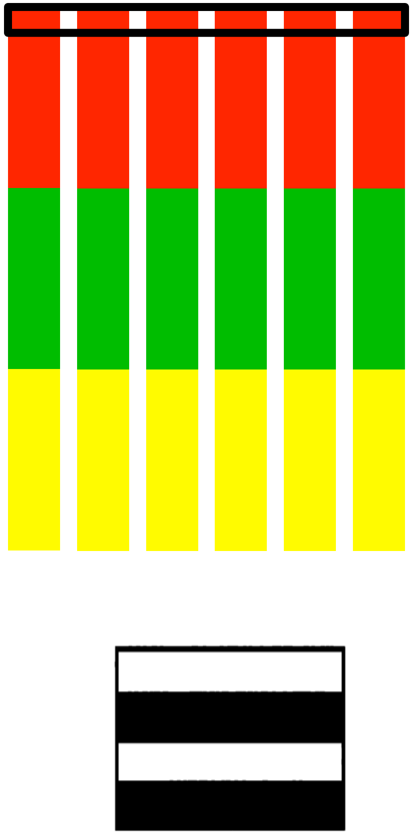
IO



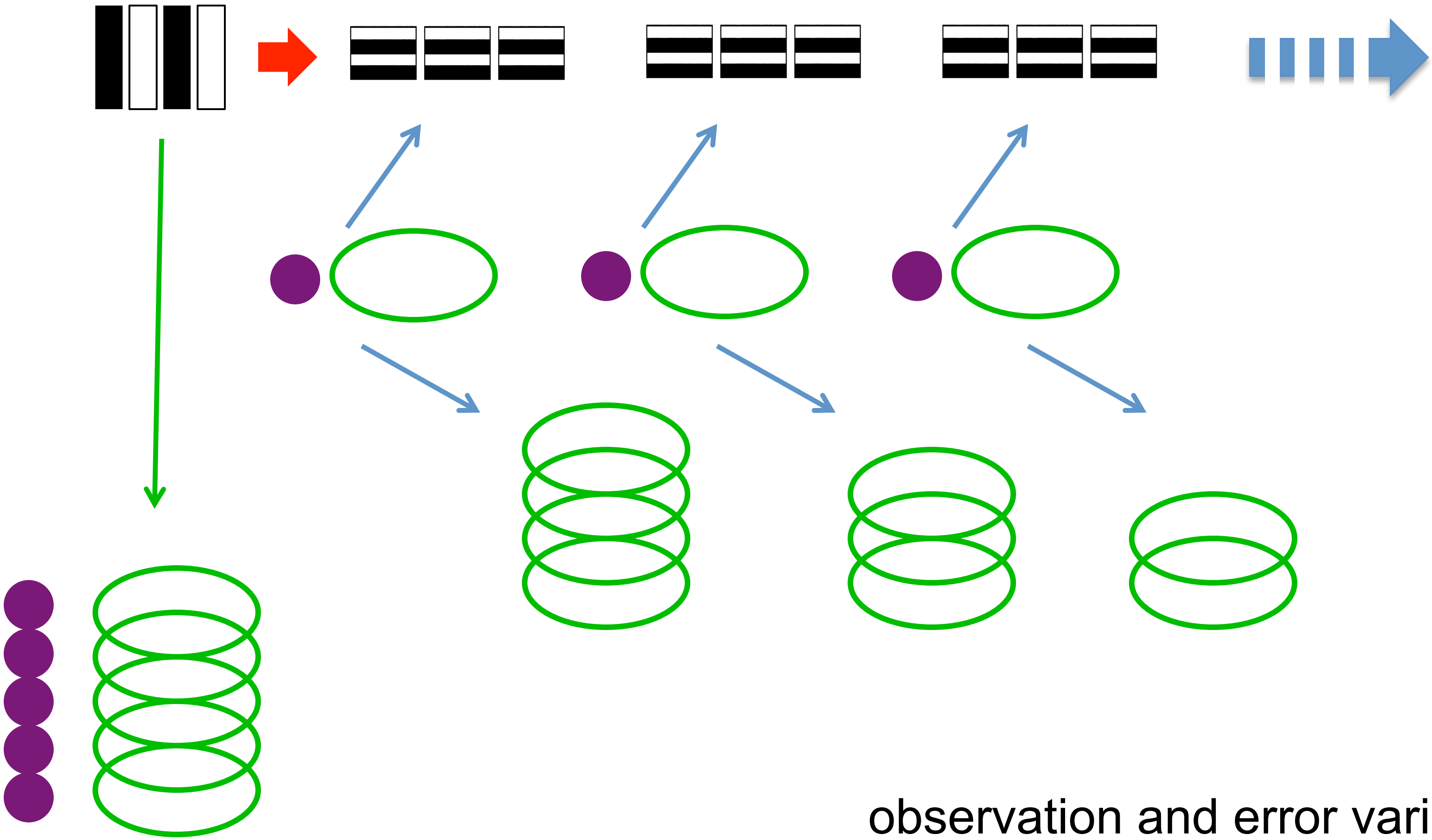
Should the IO speed drive the data layout?

Notation





What’s parallel about DART?



observation and error variance ●

ensemble approximation of the observation ○

updates →

Why do we need to change anything?

Or, what’s not so parallel about DART?

- Multiple data decompositions
- IO
- Algorithm choice and communication

Limitations of having these two decompositions:

The forward operator does not scale beyond processors = ensemble members

Users have models that are too large to fit into the memory of a single node

You have to transpose data between decompositions