4D-Var: Optimisation and performance on the NEC SX-6

ECMWF presentation - 27th October 2004

Dr Stephen Oxley

Introduction

Talk Outline

- Porting Data Assimilation codes to the SX-6
- Optimisation techniques and impacts
- Effect of PE domain decomposition
- Code scalability
- Additional operational considerations

Porting Data Assimilation codes to the SX-6

Porting DA codes to the SX-6

- Non Standard/Ambiguous use of Fortran:
  - Illegal usage of structures (OPS)
  - Unformatted internal writes (OPS/VAR)
  - Expand out implied-do loop internal writes

- Functionality Changes
  - Replacement of NAG libraries: Now use Lapack-blas (OPS/VAR)
  - Reduce size of MPI ID tags and terminate correctly
  - Allow more LS States than PEs (VAR)

Porting DA codes to the SX-6

- C – Fortran Interface:
  - Ensure correctly declared sizes between routines.
- Optimisation Level:
  - Change level until gives correct results with highest optimisation (Obvious)
  - For time critical routines, track down individual loops that prevent using higher optimisation levels and deal with them.

Optimisation Techniques and Impacts

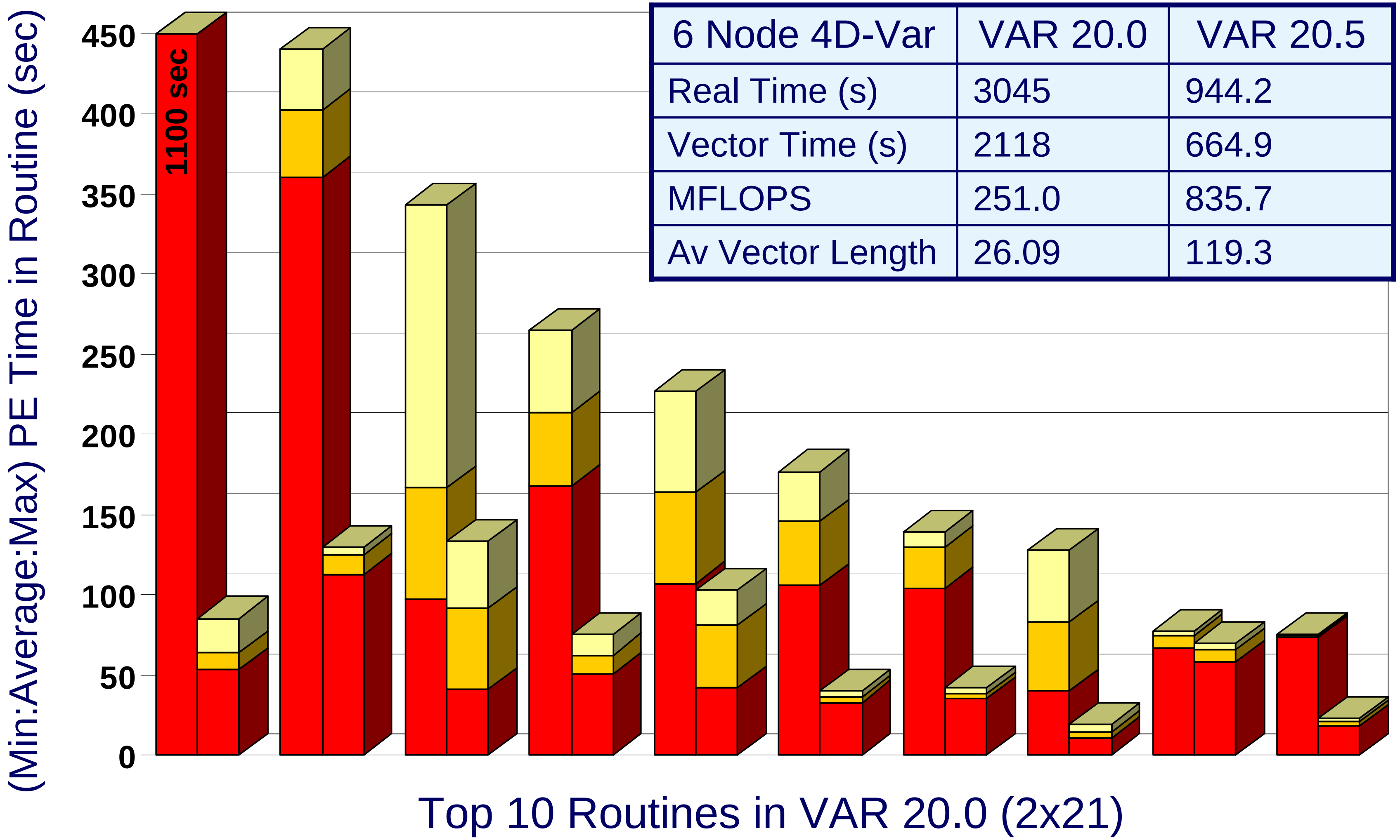
Optimisation Techniques and Impacts

- On SX-6 most important to:
  - Ensure loops vectorise well
  - Reduce memory bank conflicts
  - Reduce load imbalance
  - Make efficient use of IO

Optimisation Techniques and Impacts

- Standard test job – similar to what is operational:
  - 4D-Var N108L38 (216x163x38) 6 hour assimilation window with 20 minute time step => 18 Linearisation States.
  - Converges to a solution which minimises the difference between the LS States and observations in ~60 iterations.
  - In 4D-Var each of the 60 iterations performs a forward forecast and its adjoint (essentially a backward forecast)
  - These forecast routines account for practically all of the run-time and is why 4D-Var is an order of magnitude more demanding than 3D-Var

Overall Performance: 4D-Var Version 20.0 vs 20.5



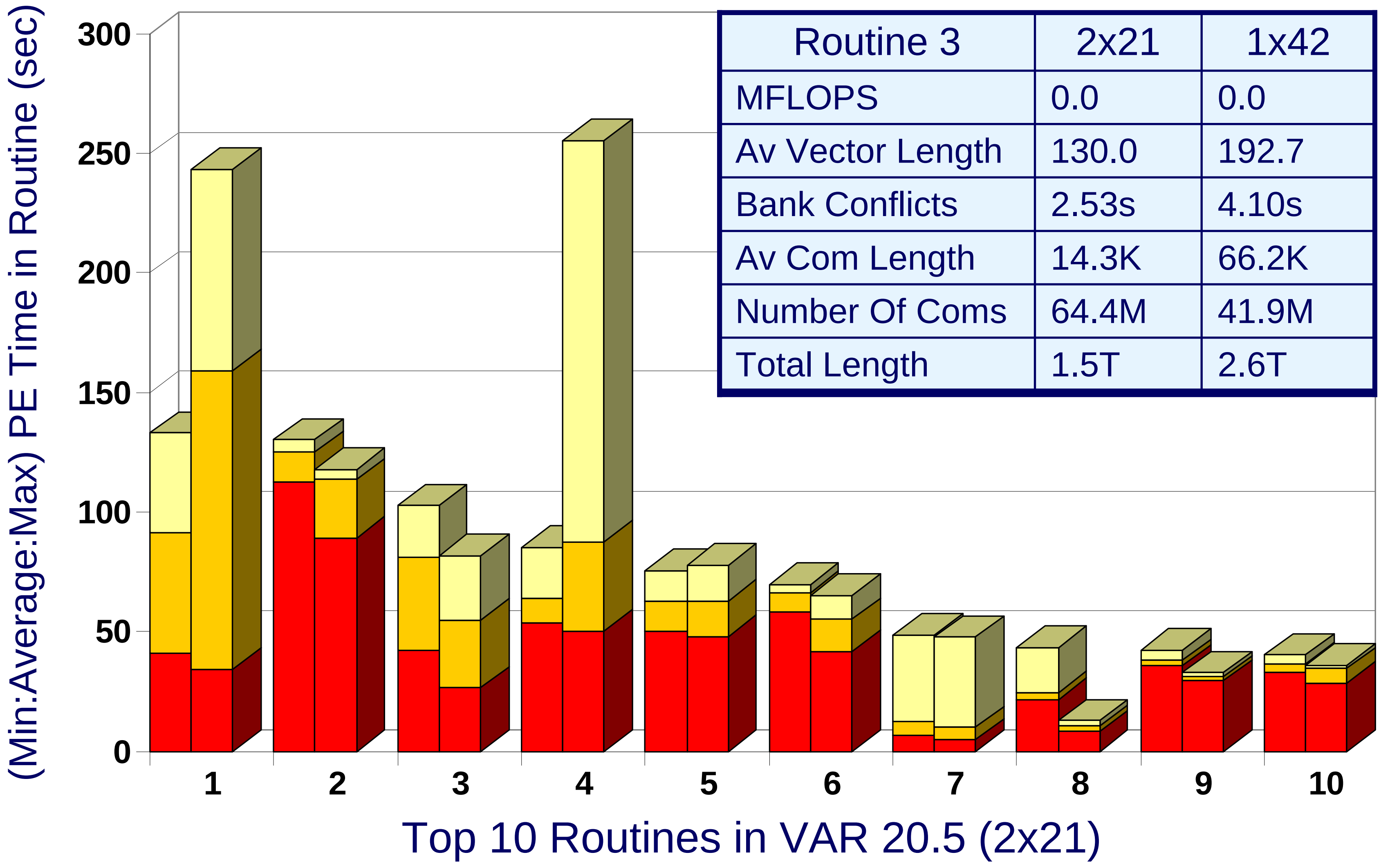
6 Node 4D-Var	VAR 20.0	VAR 20.5
Real Time (s)	3045	944.2
Vector Time (s)	2118	664.9
MFLOPS	251.0	835.7
Av Vector Length	26.09	119.3

Most Popular Optimisation Techniques Applied (Number of routines in Top 10 applied to)

1. Remove unnecessary MPI summation orders (5)
2. Loop splitting to remove dependence of calculations (5)
3. Change loop nest order to aid vectorisation (4)
4. Collapse loops by hand to increase vector lengths (3)
5. Loop merging (3)
6. Use additional work arrays to avoid bank conflicts (2)

Effect of PE Domain Decomposition

Effect of PE Decomposition: 4D-Var on 6 Nodes



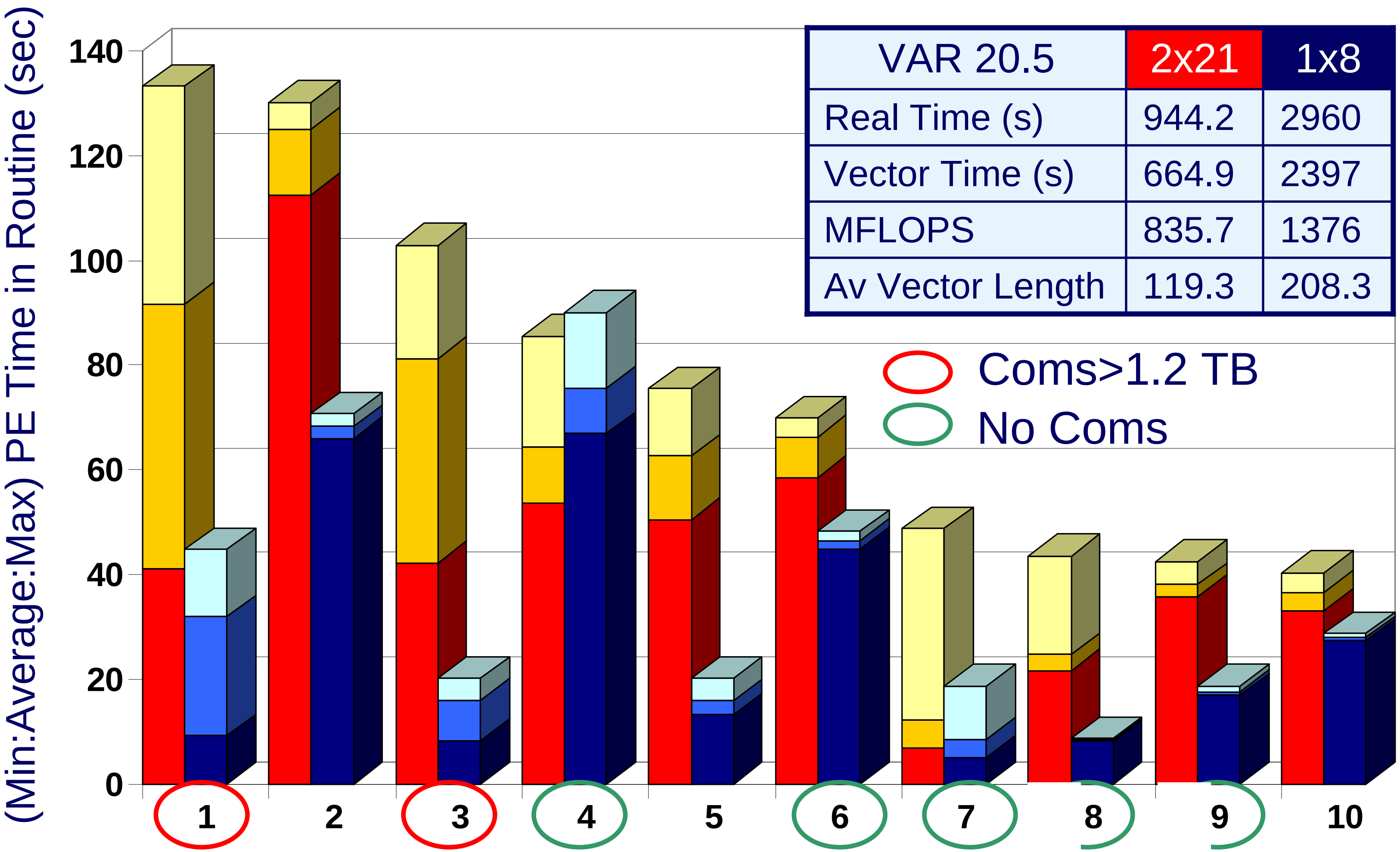
Routine 3	2x21	1x42
MFLOPS	0.0	0.0
Av Vector Length	130.0	192.7
Bank Conflicts	2.53s	4.10s
Av Com Length	14.3K	66.2K
Number Of Coms	64.4M	41.9M
Total Length	1.5T	2.6T

Code Scalability

Code Scalability: 4D-Var 6 Nodes vs 1 Node

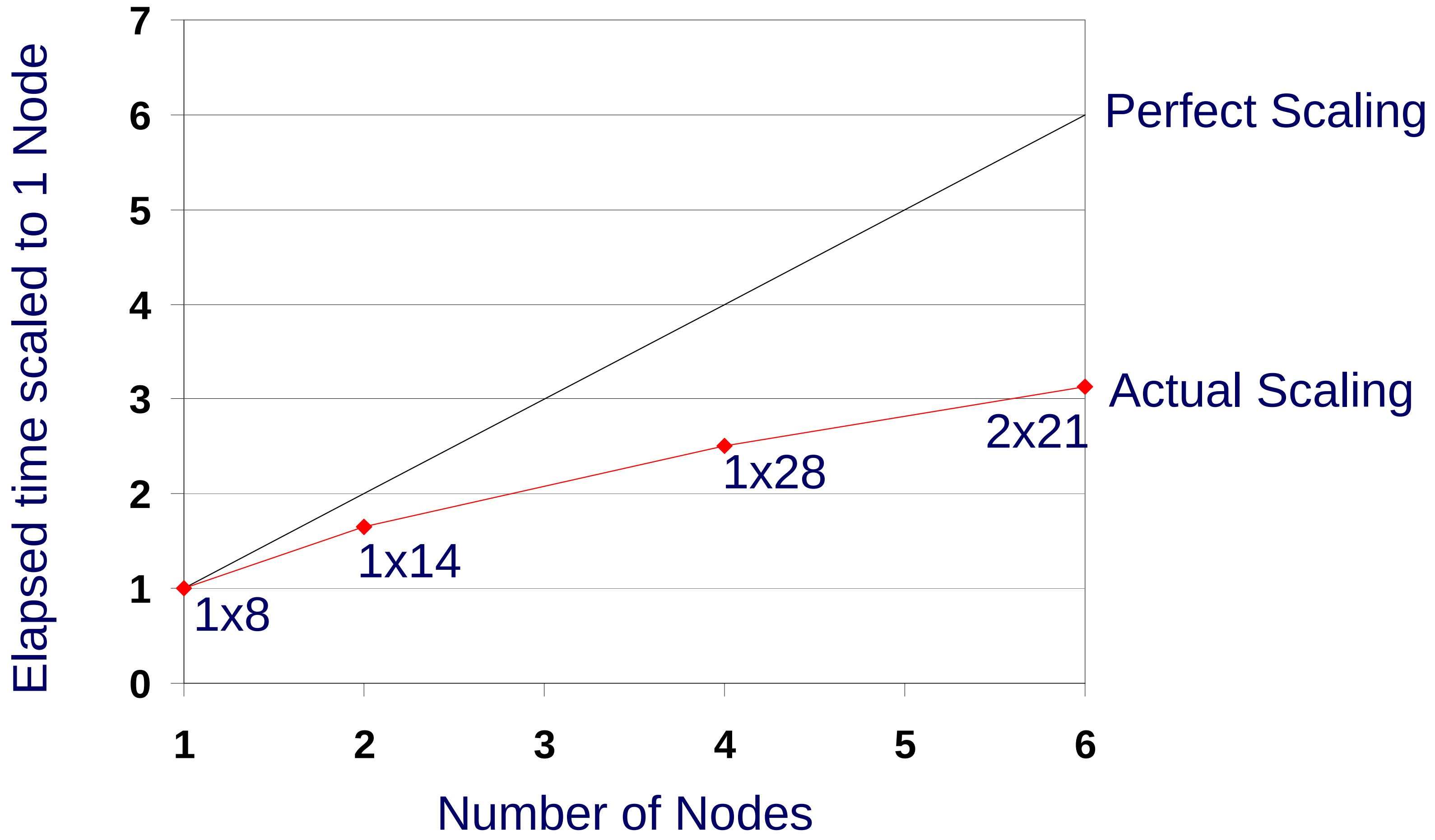
1 node times are scaled to 6 node

VAR 20.5	2x21	1x8
Real Time (s)	944.2	2960
Vector Time (s)	664.9	2397
MFLOPS	835.7	1376
Av Vector Length	119.3	208.3



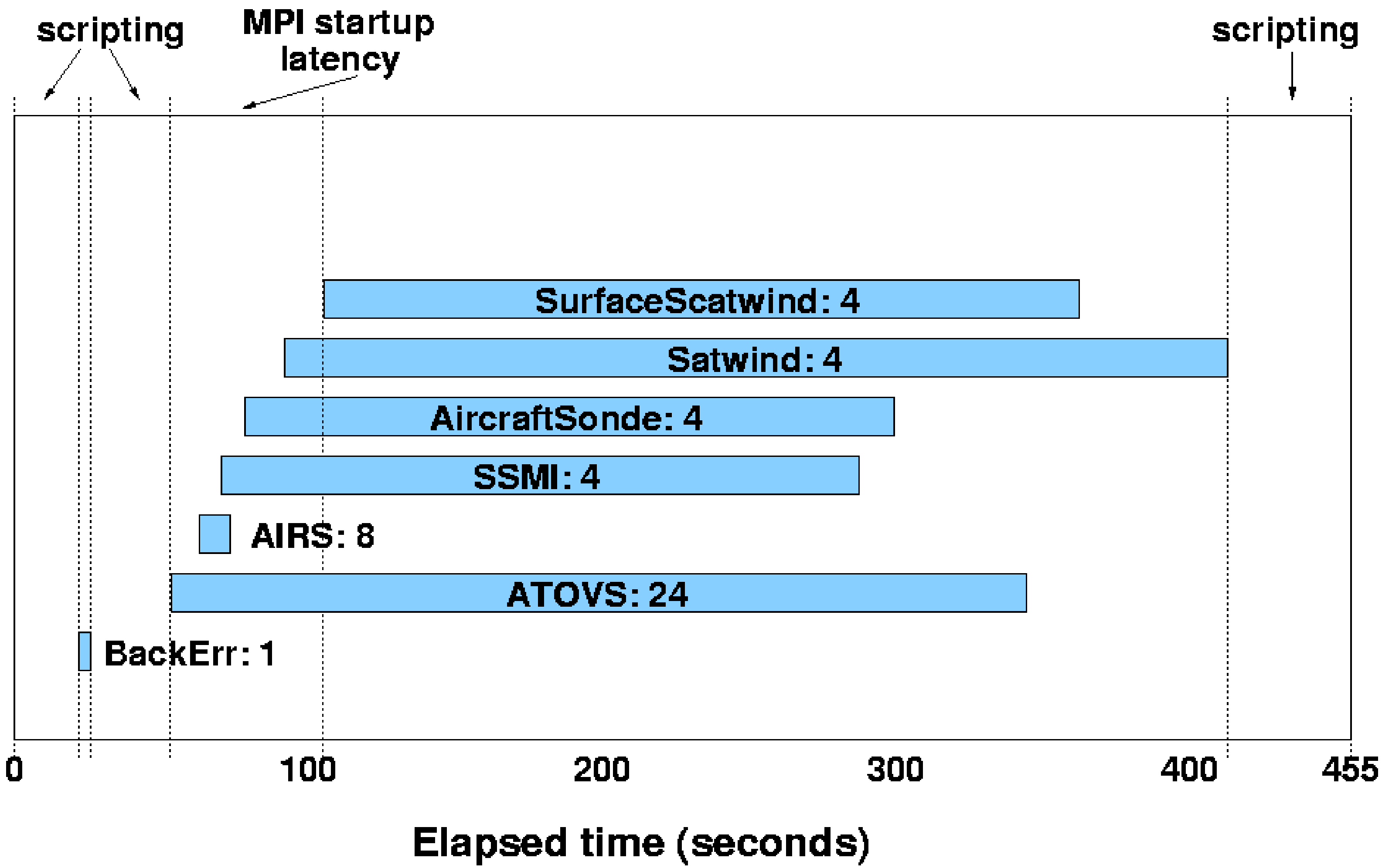
Top 10 Routines in VAR 20.5 (2x21)

Scaling of 4D-Var N108L38



Additional Operational Considerations

Observation Processing Operational Timings



Questions & Answers

Top 10 VAR 20.0 at 2x21

1. Cubic_lagrange_adj
2. Gcr_elliptic_operator_adj2
3. Swap_bounds_adj
4. Gcr_coefficient
5. Swap_bounds
6. Gcr_elliptic_operator
7. Mpp_tri_solve_exec
8. Gcr_precon_adi_exec_tri_solve
9. Vert_weights
10. Gcr_precon_adi_exec_tri_adj2

Optimisation Techniques and Impacts

	Routine Name/Opt Technique Num	1	2	3	4	5	6
1	Cubic_lagrane_adj		X	X	X	X	X
2	Gcr_elliptic_operator_adj2	X	X			X	
3	Swap_bounds_adj			X			
4	Gcr_coefficient	X					
5	Swap_bounds						
6	Gcr_elliptic_operator	X	X			X	
7	Mpp_tri_solve_exec		X	X	X		X
8	Gcr_precon_adi_exec_tri_solve	X	X				
9	Vert_weights				X		
10	Gcr_precon_adi_exec_tr_adj2	X		X			

Top 10 VAR 20.5 at 2x21

1. Swap_bounds_adj
2. Gcr_elliptic_operator_adj2
3. Swap_bounds
4. Cubic_lagrange_adj
5. Gcr_coefficient
6. Vert_weights
7. Ritchie
8. Interpolation
9. Mpp_tri_solve_exec
10. Gcr_elliptic_operator