Recent performance improvements in HIRLAM 4D-VAR

Tomas Wilhelmsson

Swedish Meteorological and Hydrological Institute

HIRLAM All Staff Meeting

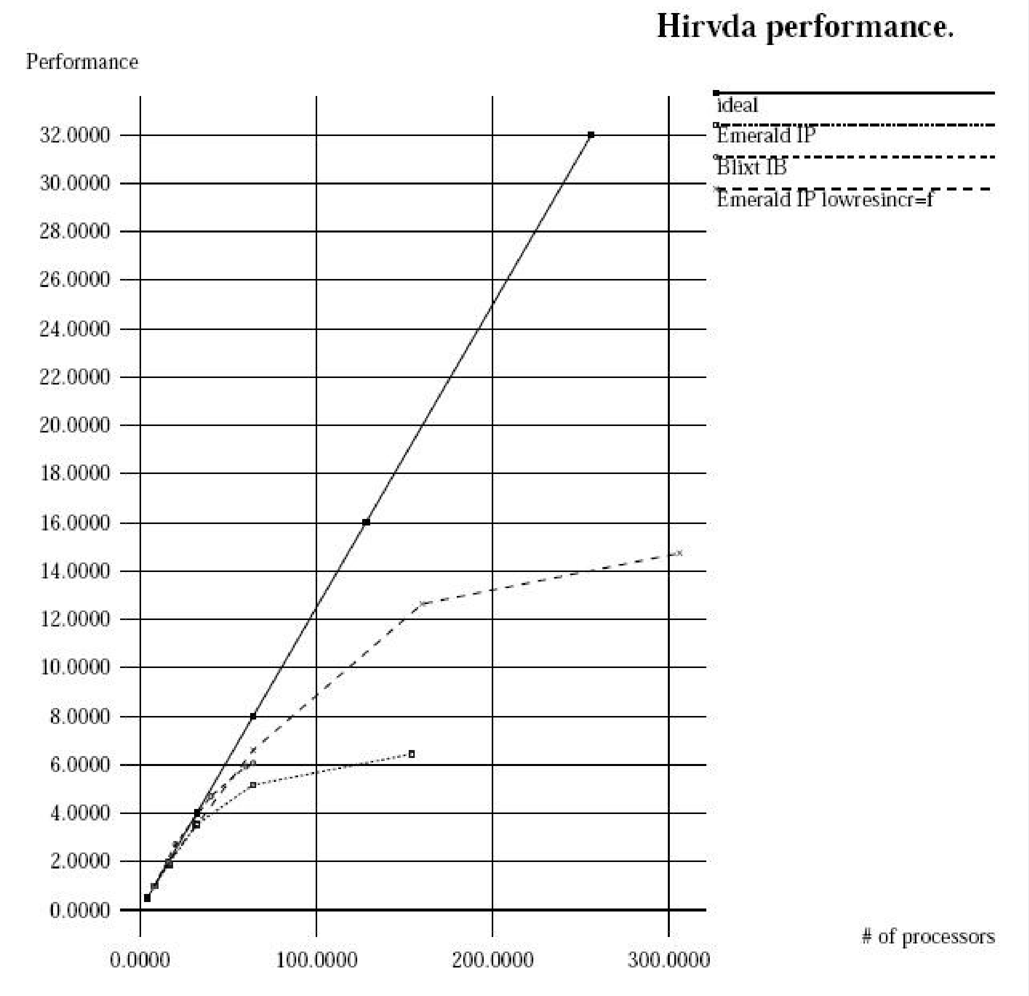
2008-04-09

HIRVDA 4D-VAR performance scaling

Niko Sokka (FMI):

> ***"In the end of day, 4DVAR scales up to 84 processors in our system and then stalls."***

Torgny Faxén (NSC):



Signatur

Room for improved scaling

- **Additional sources for parallelism**
  - **OpenMP**

- **Reduce interprocessor communication**
  - **Slswap on demand**

- **Reduce work**
  - **Fewer FFTs   (also reduces communication)**

Moore's law

- “Computer performance doubles every 18-24 months”

- What Gordon Moore really said (1965 and 1975):
  - *The number of transistors that cheapest can be integrated on a chip doubles every two years.*

- Until 2005 processors really got faster, since smaller transistors can also be clocked faster
  - But now clock frequencies are limited due to power and heat dissipation
  - However, the number of transistors on a chip still increases

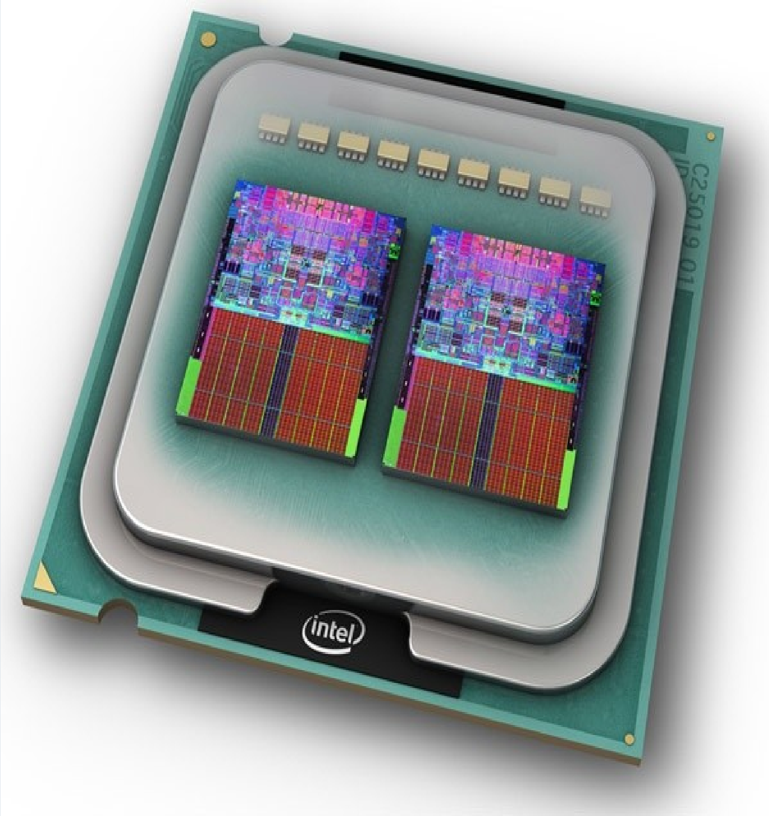
- Instead of faster processors, we now get more processors per chip

Multi-core processors

- Is your new laptop “dual-core”?

- Multi-core is everywhere:
  - Linux clusters
  - SGI Altix
  - IBM PowerPC
  - Cray XT

- SMHI's new cluster will use Intel “Harpertown” Quad-core chips:



Intel Clovertown performance compared to current SMHI clusters

- **Clovertown dual socket quad core at 2.66 GHz (8 processors)**
- **Dunder dual socket single core at 3.4 GHz (2 processors)**
- **Nodes are three times faster, performance per processor has decreased!**
- **HIRVDA speedup??**

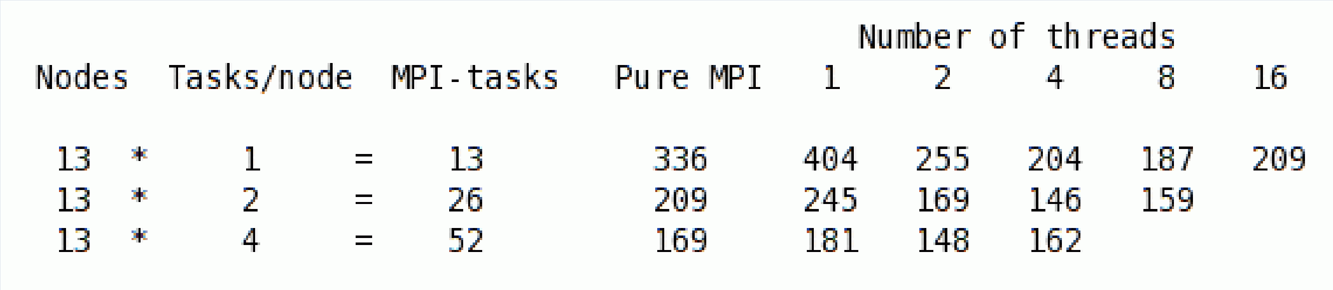
Model	Clovertown vs. Dunder performance
HIRLAM	3.3
HIRVDA	3.1
AROME	3.6

OpenMP to the rescue for HIRLAM 4DVAR?

- **Multi-core nodes lends themselves to shared memory parallelization with OpenMP. Quick inter-core communications**
- **Larger MPI tasks, fewer, larger messages**
- **Incremental approach**
- **Physics was already done (through `phcall` and `phtask`)**
- **Now Semi-Langrangian dynamics too**

OpenMP performance

- SMHI C22 area, 306x306 grid
  - Inner TL/AD loop at 1/3 resolution, 103x103 grid
  - What could OpenMP give on HPCE?



						Number of threads				
Nodes		Tasks/node		MPI-tasks	Pure MPI	1	2	4	8	16
13	*	1	=	13	336	404	255	204	187	209
13	*	2	=	26	209	245	169	146	159	
13	*	4	=	52	169	181	148	162		

- OpenMP “works”, but doesn't improve much over pure MPI for this case

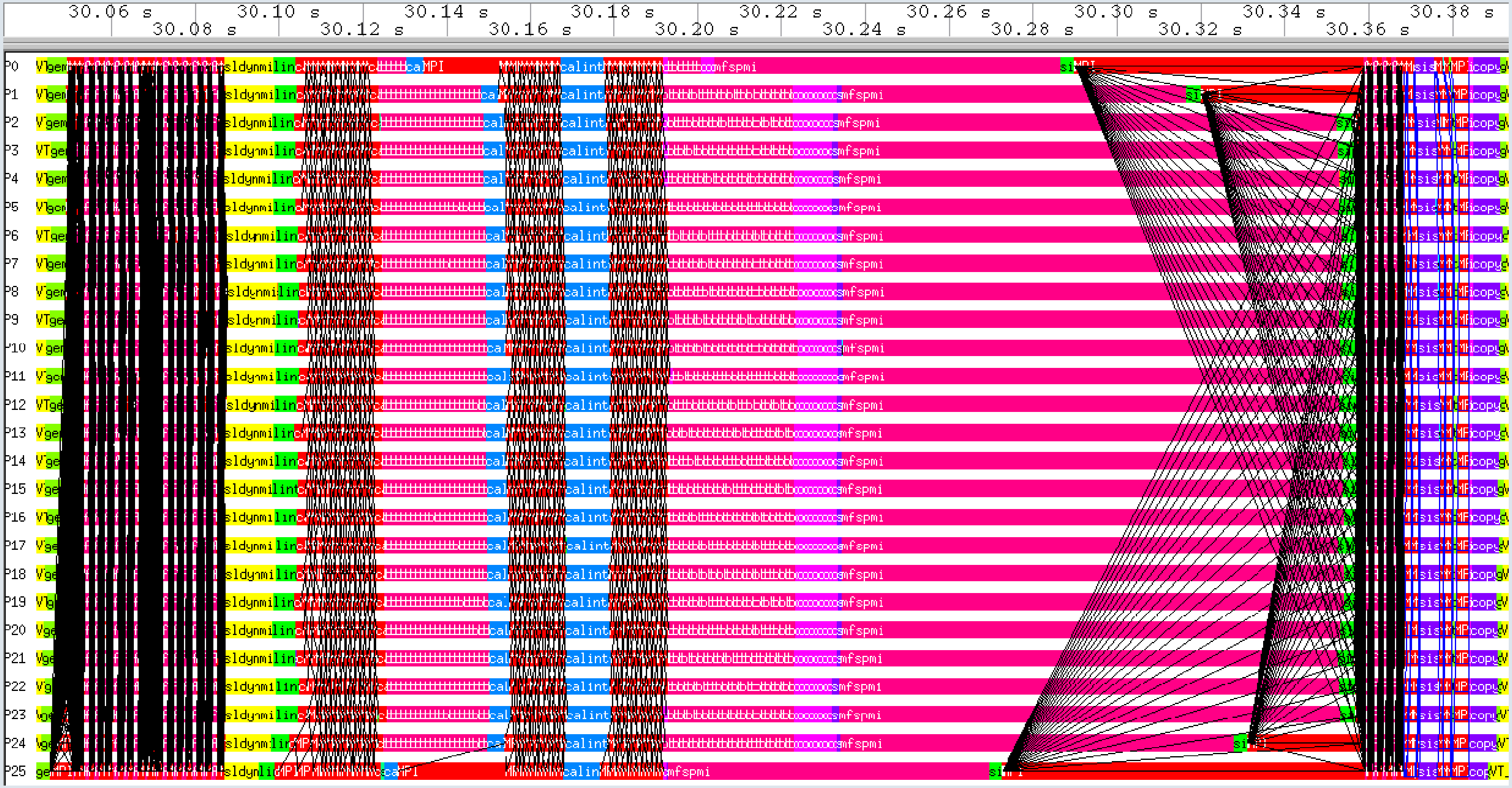
- Note: `LEN_LOOPS` determines tasking in `phcall`.
  - Default was 2047, but 16 is better for HPCE

OpenMP problems

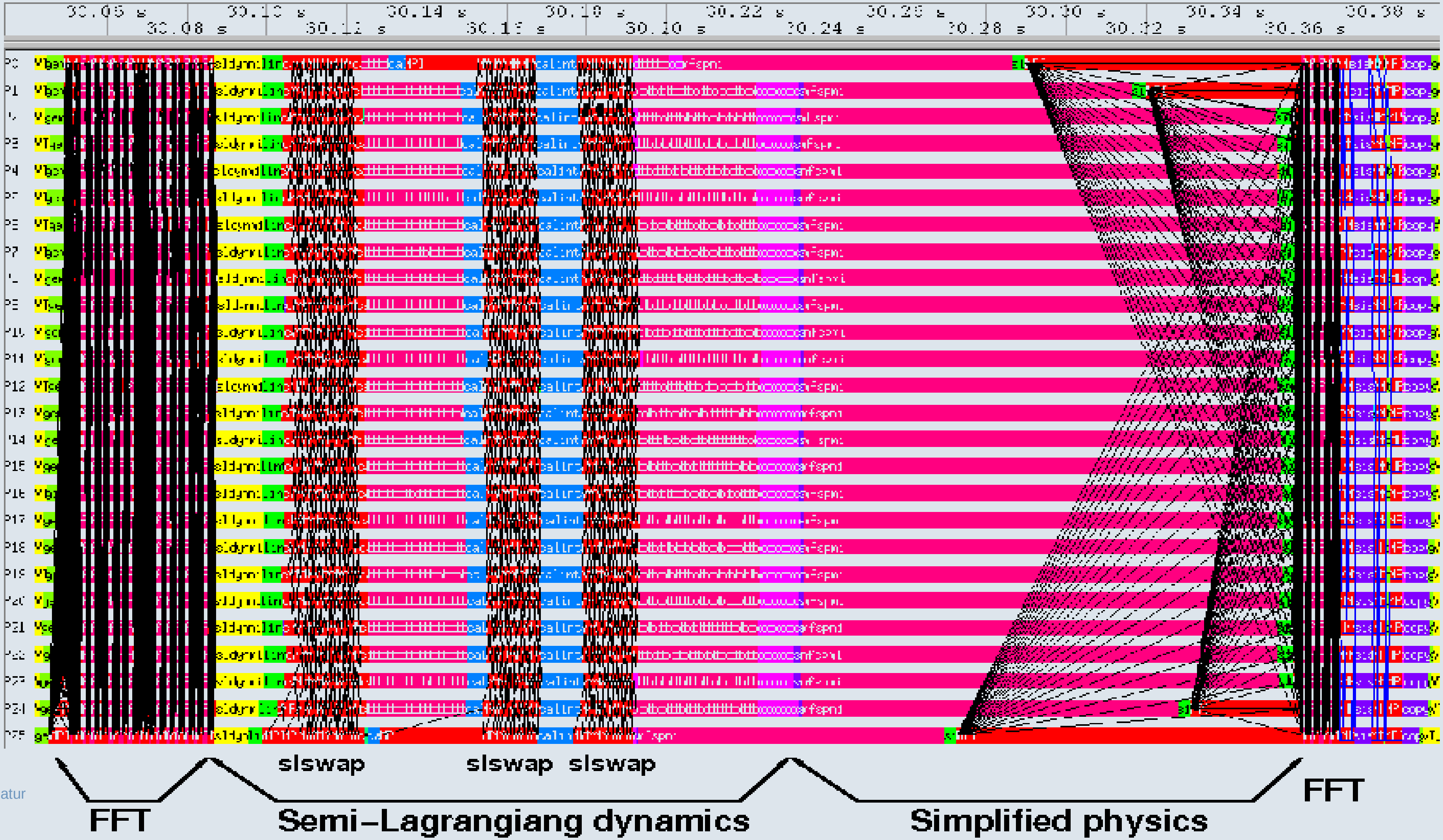
- The larger 582x448 RCR area would be more interesting
  - Inner loop at half resolution, 292x225 grid
  - But with OpenMP it crashes around `RTMIOSYS`

- Cannot combine (Scali) MPI with Intel Fortran OpenMP on current SMHI clusters
  - Severe slowdown if combined, but either MPI or OpenMP works fine

HIRVDA TL time step on 26 processors

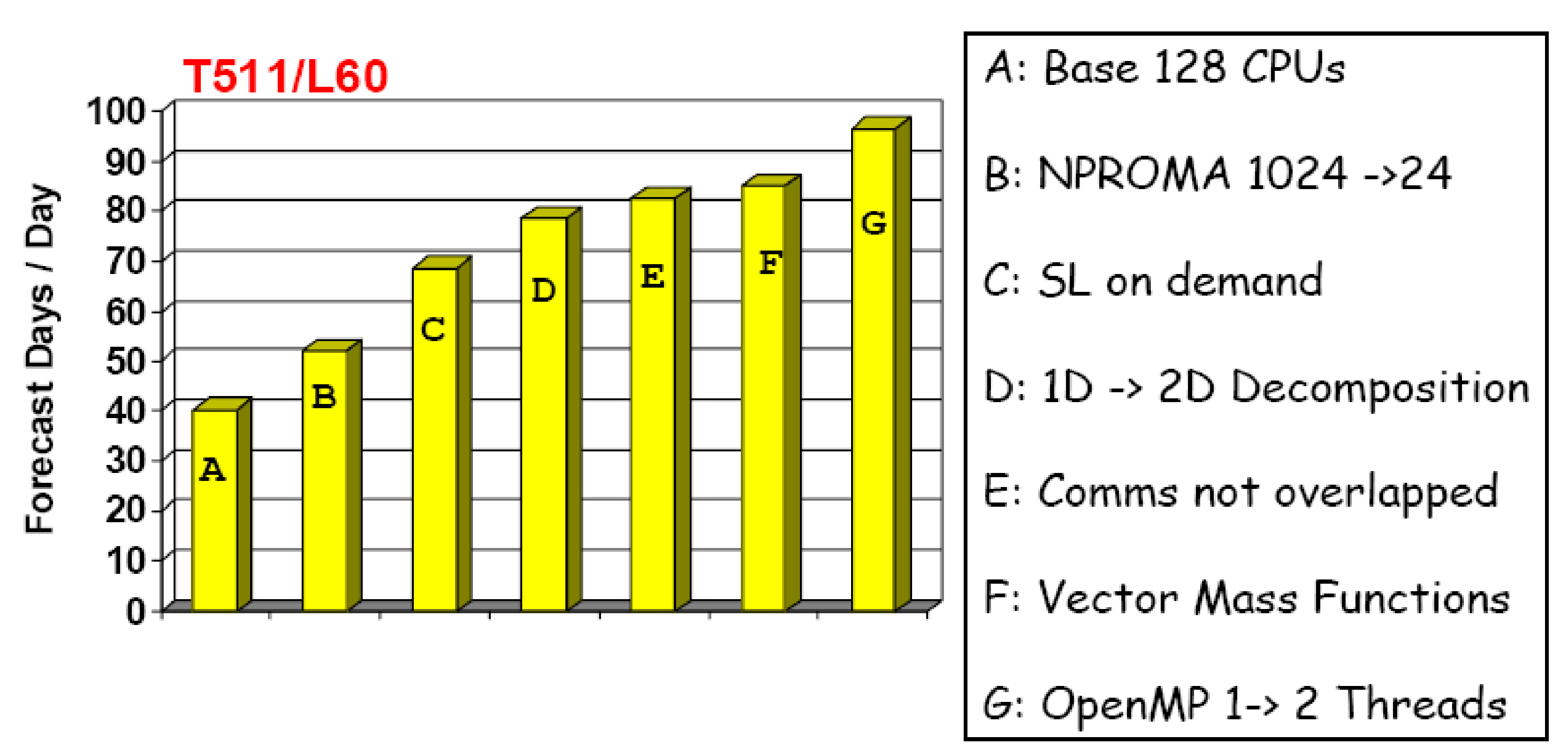


HIRVDA TL time step on 26 processors



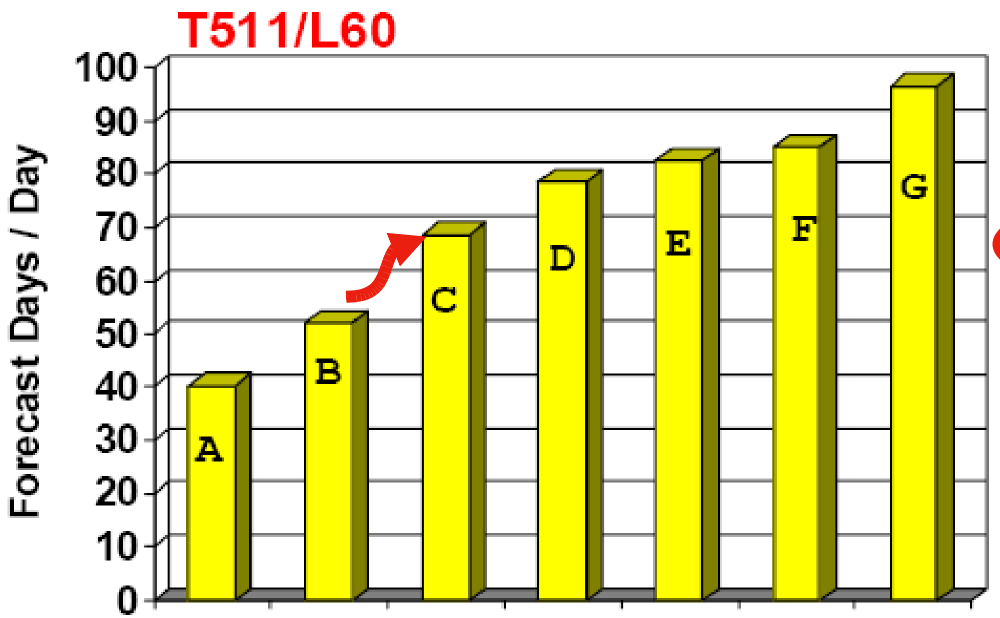
Signatur

Effects on various optimizations on IFS performance (Debora Salmond, 2002)



Moving from Fujitsu VPP (vector machine) to IBM SP (cluster).

Effects on various optimizations on IFS performance (Debora Salmond, 2002)



A: Base 128 CPUs

B: NPROMA 1024 ->24

C: SL on demand

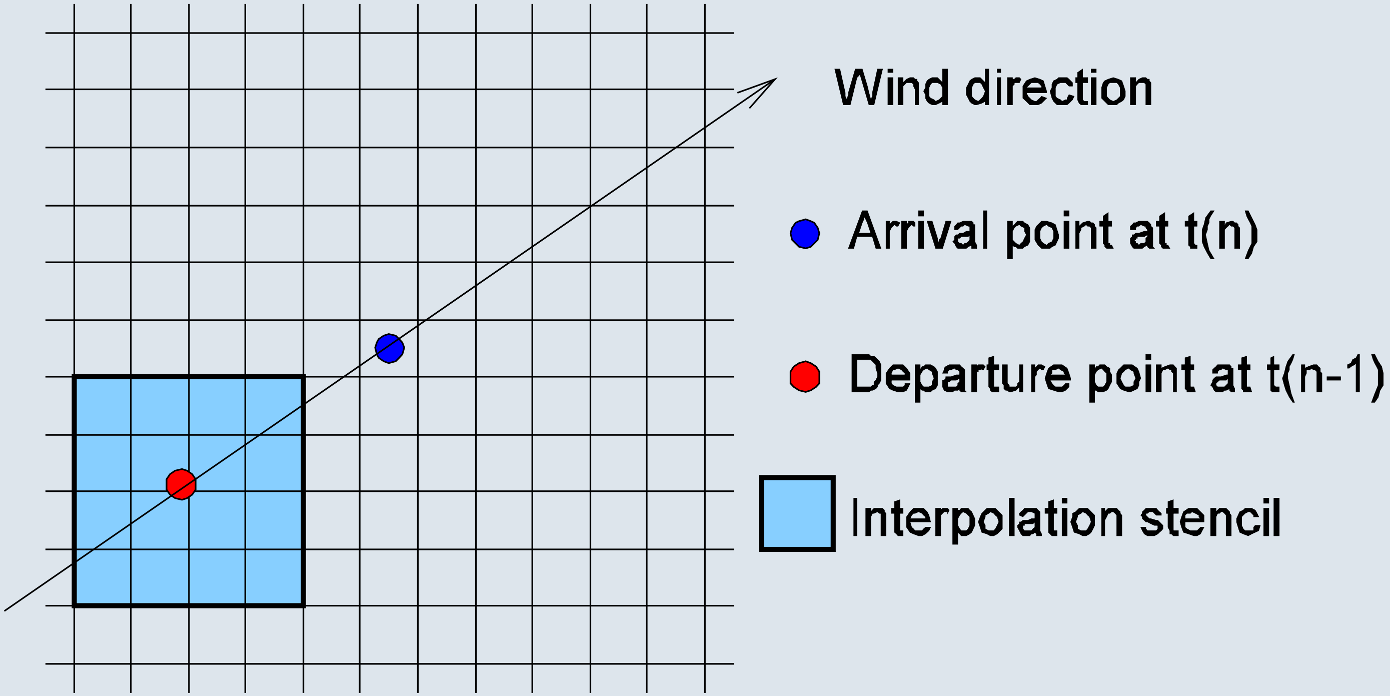
D: 1D -> 2D Decomposition

E: Comms not overlapped

F: Vector Mass Functions

G: OpenMP 1-> 2 Threads

Semi-Lagrangian Advection



Wind direction

🔵 Arrival point at t(n)

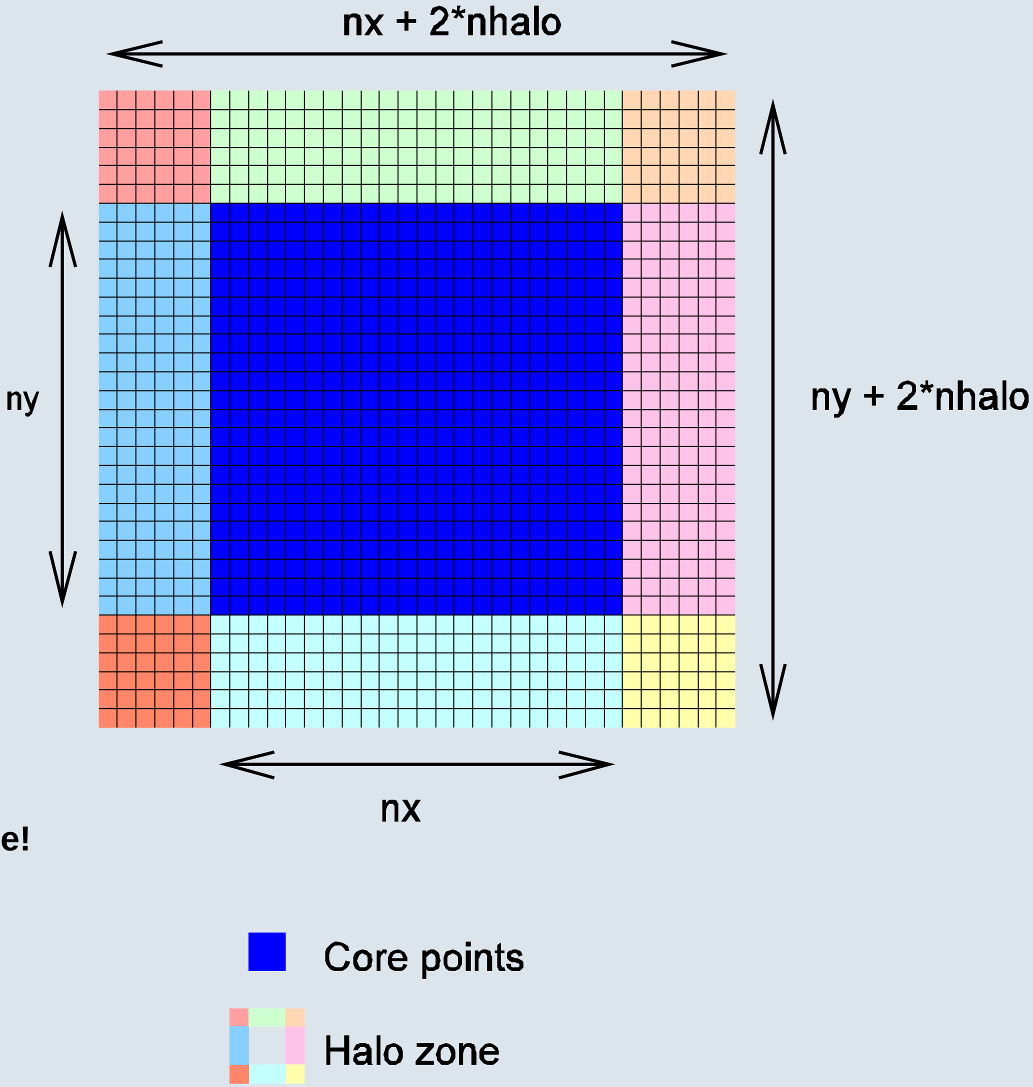
🔴 Departure point at t(n-1)

🟦 Interpolation stencil

- **Full cubic interpolation in 3D is 32 points (4x4x4)**

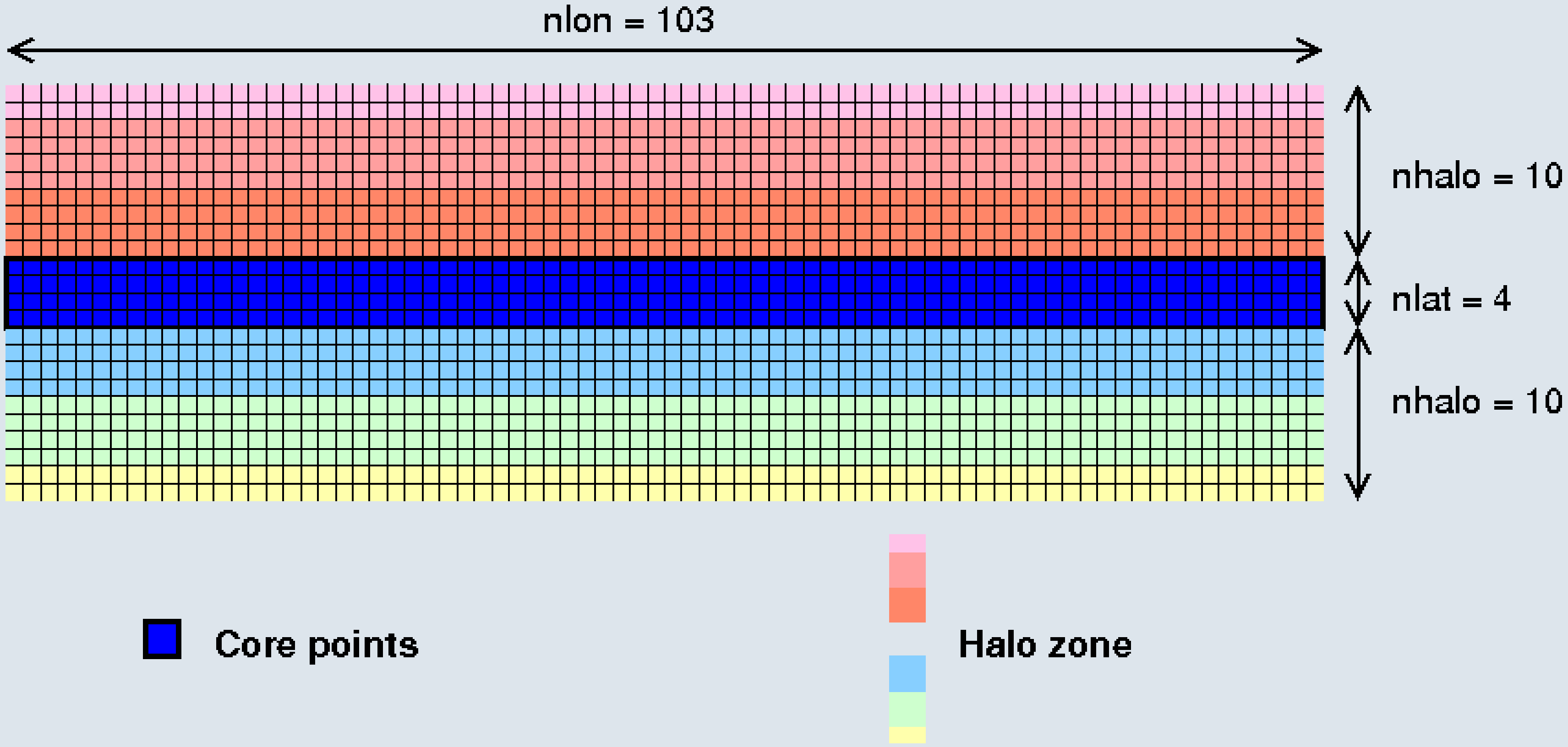
Example: The HIRLAM C22 area (306x306 grid at 22 km resolution)

- Max wind speed in jet stream 120 m/s
- Time step 600 s
- => Distance 72 km = 3.3 grid points)
- Add stencil width (2) => nhalo= 6
- With 64 processors partitioned in 8x8:
  - 38x38 core points per processor
  - 50x50 including halo
- Halo area is 73% of core!
- But full halo is not needed everywhere!

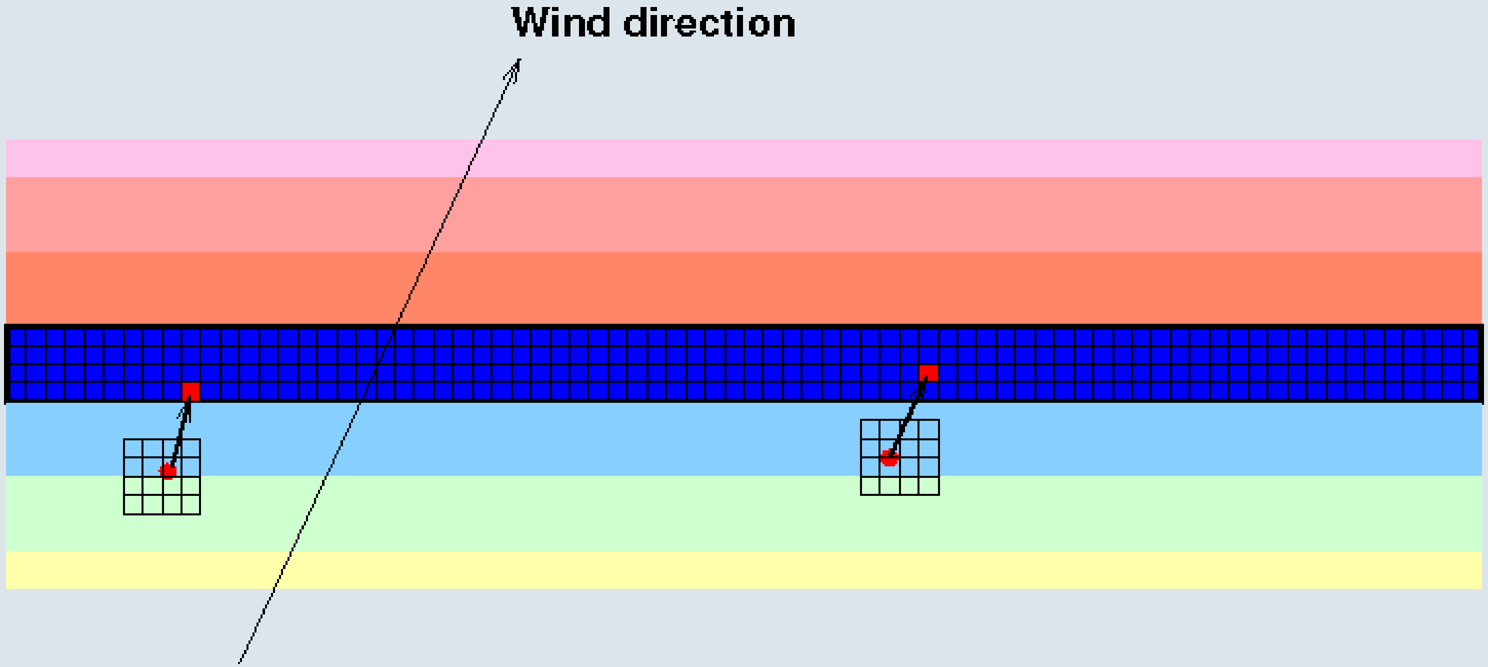


HIRVDA 4D-VAR domain decomposition

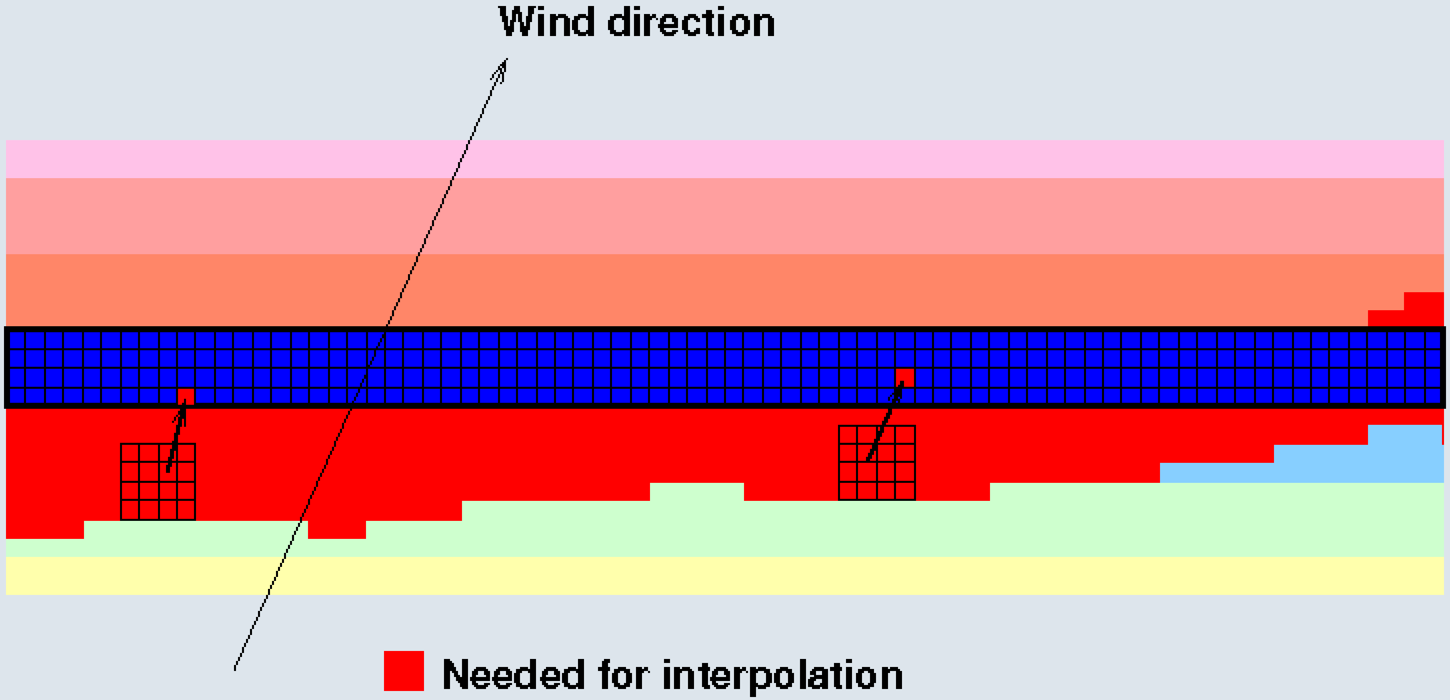
- SMHI C22 area inner loop at 1/3 resolution, with 30 minute time step
  - 103 x 103 grid distributed over 26 processors, each get a 103x4 slice



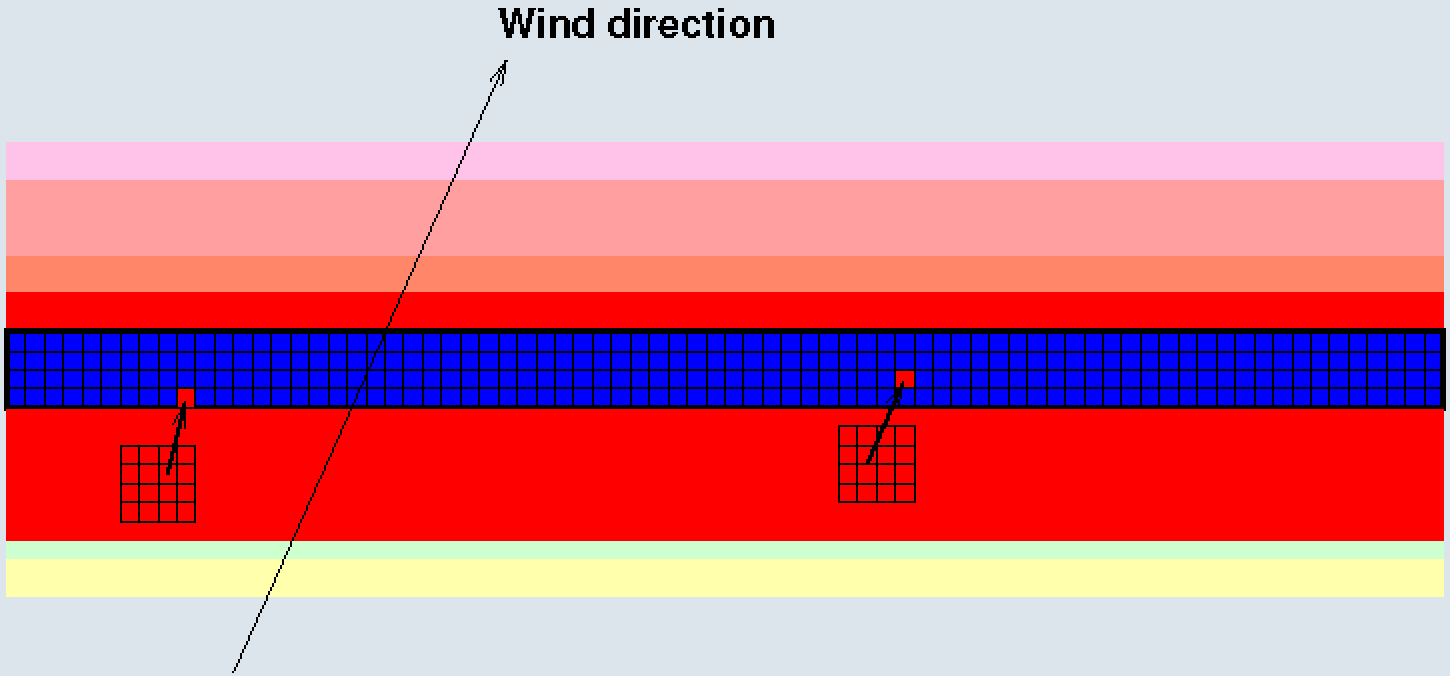
Interpolation stencils requiring halo communication



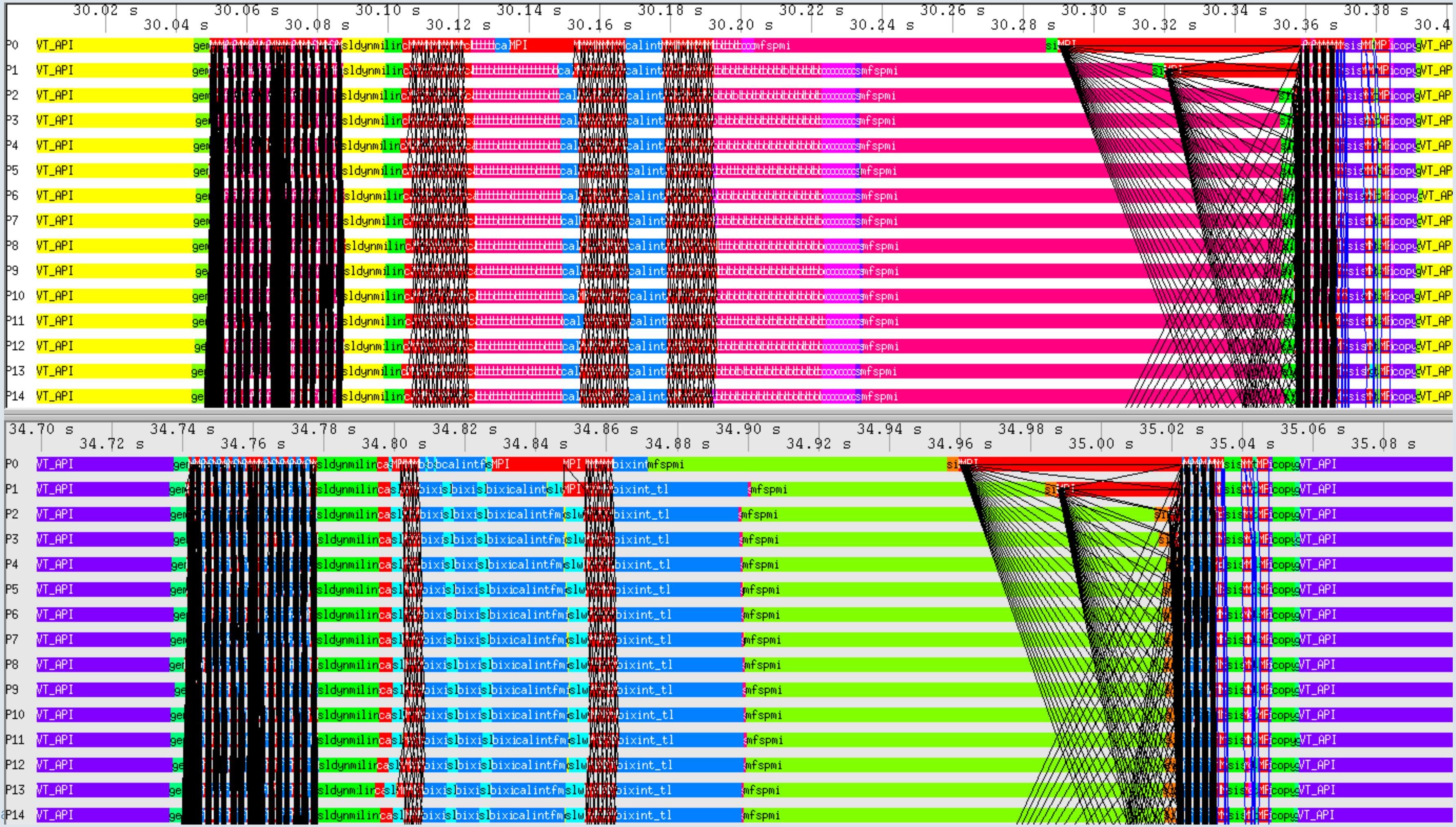
Accumulated stencils in halo zone



Communicate whole rows with at least one required point

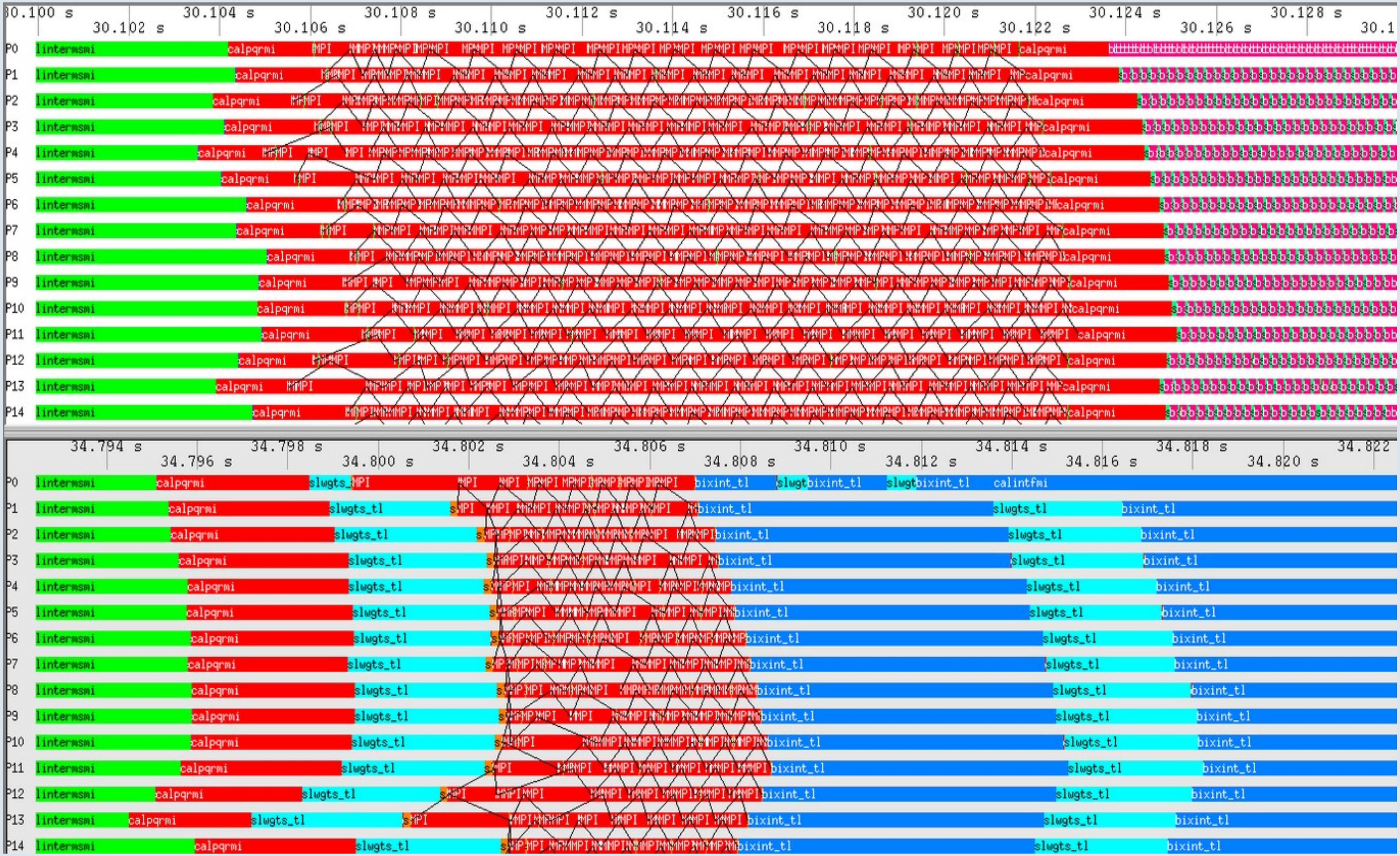


Tangent-Linear timestep without and with slswap' on demand



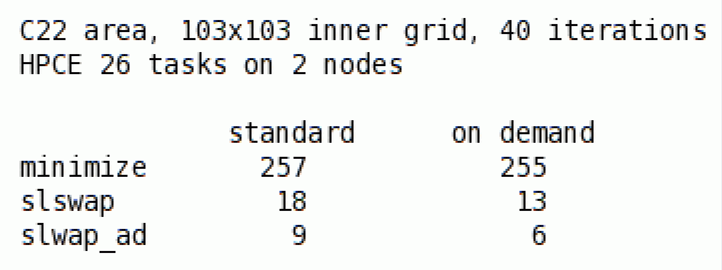
Signa

Zoom in on calpqr_tl



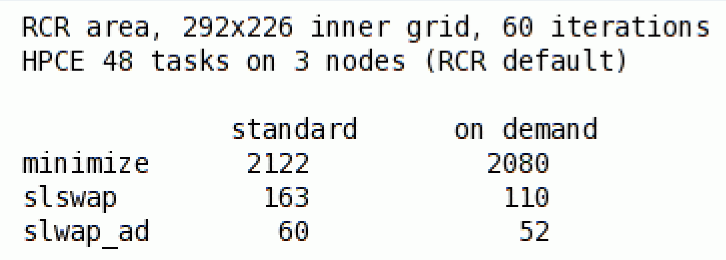
Slswap on demand timings

- **Slswap 30-50% faster**
- **No significant improvement in runtime**
- **Load balance issues?**



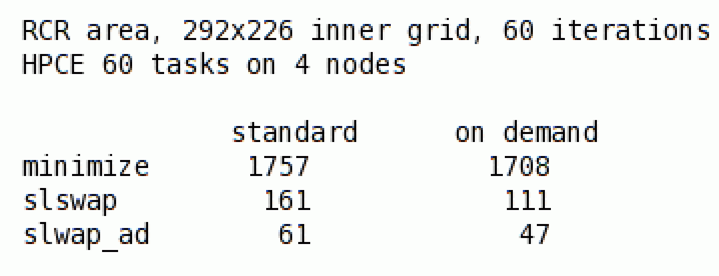
```
C22 area, 103x103 inner grid, 40 iterations
HPCE 26 tasks on 2 nodes

             standard      on demand
minimize       257           255
slswap          18            13
slwap_ad         9             6
```



```
RCR area, 292x226 inner grid, 60 iterations
HPCE 48 tasks on 3 nodes (RCR default)

             standard      on demand
minimize      2122          2080
slswap         163           110
slwap_ad        60            52
```



```
RCR area, 292x226 inner grid, 60 iterations
HPCE 60 tasks on 4 nodes

             standard      on demand
minimize      1757          1708
slswap         161           111
slwap_ad        61            47
```

Nils Gustafsson: Remove and shorten FFTs

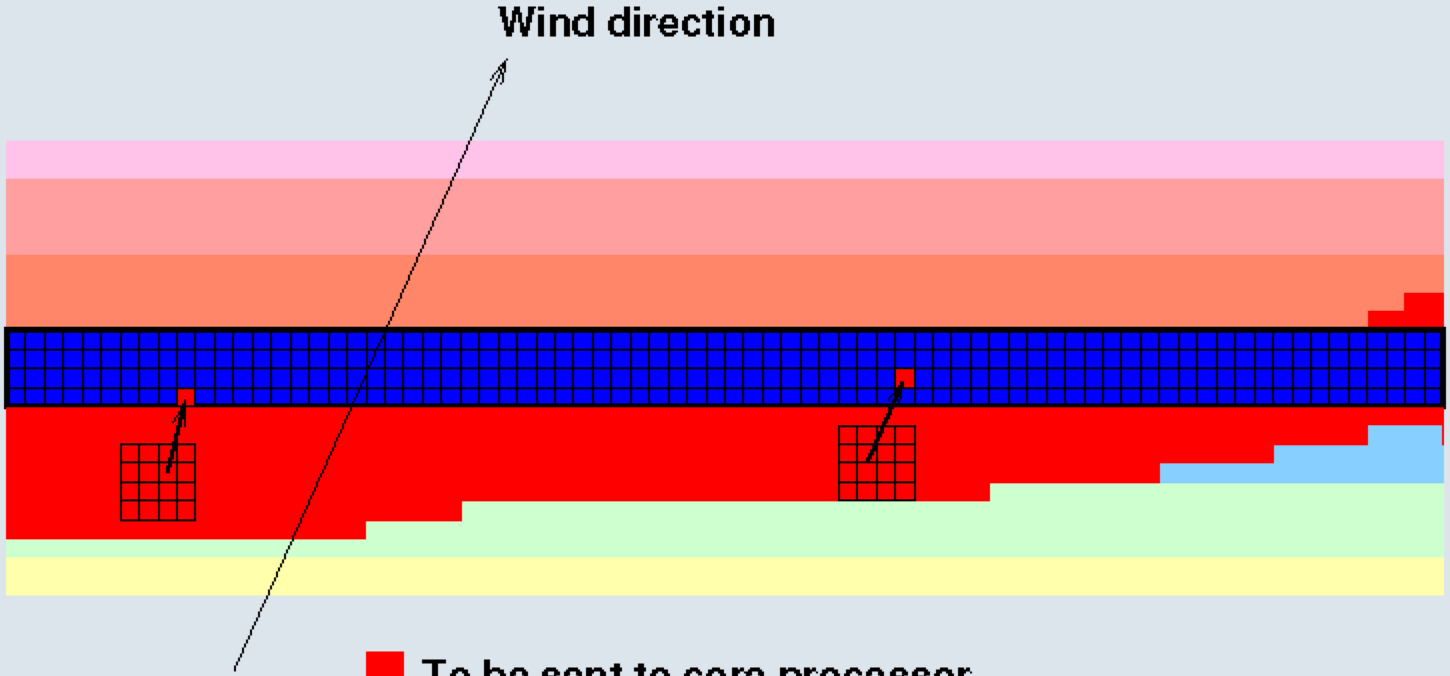
- Store trajectory in physical instead of spectral space
  - Removes 7-8 out of 25 FFTs
  - 7-8% faster!
- Smaller extension zone in TL and AD
  - Shorter FFTS (e.g. 400 down to 310)
  - Another 3-4% faster!
- Soon to be include in trunk...

Conclusion

- **HIRVDA OpenMP still not shown to be really useful**
- **Slswap on demand works, and expected to be more important for high numbers of processor**
- **Nothing beats avoiding work completely**
- **No silver bullet**
  - **Performance from small incremental improvements**

Technology will allow larger areas “for free”, but increasing the number of time steps and maintaining runtime will be harder

Expand to “continuous rows”



🟥 To be sent to core processor