CMCC Activity plan on NEMO

NEMO HPC-WG
10 Apr 2019

I.Epicoco, S.Mocavero, F.Mele, M.Chiarelli

cmcc
Centro Euro-Mediterraneo sui Cambiamenti Climatici

is-enes
INFRASTRUCTURE FOR THE EUROPEAN NETWORK FOR EARTH SYSTEM MODELLING

immerse
IMPROVING OCEAN MODELS FOR THE COPERNICUS PROGRAMME

ESCAPE 2

esiwace
CENTRE OF EXCELLENCE IN SIMULATION OF WEATHER AND CLIMATE IN EUROPE

Single node performance
Loop fusion

(IMMERSE)

- Merging all of the loops currently written inside a given kernel/scheme implementation into one bigger loop:

  We want to perform as much operations as possible for a given point (ji, jj, jk) before moving processing the next one

- Changing the layout of the matrix, that is computed inside a loop, to be compliant with the loop range
  - E.g.
    ```
    REAL :: pta (nldi:nlei, nldj:nlej, jpk) ….
    DO jk=1, jpk-1 ; DO jj=nldj, nlej; DO ji=nldi, nlei
    ```

- Avoiding any conditional statements within the loop to enhance the vectorization

- Introducing the use of pre-compiler macros to make the code more readable and maintainable

MPI Communication

(IMMERSE, IS-ENES3)

- Moving the MPI communication before the kernel and introducing an extended halo region
  - Reducing the number of messages sent, increasing the message size
- Investigating about the adoption of neighbors collective MPI3 calls
- Overlap communication and computation

DSL Approach

(ESCAPE2, ESiWACE2)

- Evaluation of the GridTool DSL approach by means of Dwarves development.
- Tracers’ Advection: MUSCL, TVD
- Tracers’ Diffusion: LDF
- Evaluation
  - Performance
  - Portability
  - Usability
- Definition of a NEMO test case to compare GridTool and PSyClone-CLAW DSLs

Machine Learning Investigation

(IS-ENES3, ESiWACE2)

- Evaluating new disruptive approach such as machine learning techniques to improve:
  - Estimation of parametrized phenomena
  - Downscaling
  - Data assimilation
  - Mixed model implementation
- Evaluating new computational architectures based on FPGAs

Performance Evaluation

(IS-ENES3, IMMERSE)

- Performance evaluation of coupled model as contribute to the metrics for CMIP6
- Performance Evaluation of XIOS at high resolution

Impact on NEMO code

- Developments planned for the 2019 NEMO release
  - Loop fusion (for the kernels used in our GLOB16 configuration)
    - Since the loop fusion requires radical changes of the kernel code, we propose to proceed gradually in the trunk, with strong coordination with those teams who planned to modify the same kernel. We would avoid to have a big merge at the end of the year, we prefer to work jointly. (We will discuss in the next System Team webex)
  - Extra halo
    - lbc_lnk is enriched with an extra argument to choose the number of halo lines to be exchanged
  - MPI3 neighbors collective communications
- Investigation activities
  - DSL adoption
  - Machine Learning algorithms for NEMO

BaseLine

```
DO jn = 1, kjpt             !==  loop over the tracers  ==!
   !!-- initial slop
   DO jk = 1, jpkm1
      DO jj = 1, jpjm1
         DO ji = 1, fs_jpim1
            initial_slop_i(zwx(ji,jj,jk), ji)
            initial_slop_j(zwy(ji,jj,jk), jj)
         END DO
      END DO
   END DO

   CALL mpp_lnk_3d(zwx, 1); CALL mpp_lnk_3d(zwy, 1)

   !!-- Slopes of tracer
   DO jk = 1, jpkm1
      DO jj = 2, jpj
         DO ji = fs_2, jpi   ! vector opt.
            tracer_slop(zslpx(ji,jj,jk), zwx(ji,jj,jk), zwx(ji-1,jj,jk) )
            tracer_slop(zslpy(ji,jj,jk), zwy(ji,jj,jk), zwy(ji,jj-1,jk) )
         END DO
      END DO
   END DO
   !!-- Slopes limitation
   DO jk = 1, jpkm1
      DO jj = 2, jpj
         DO ji = fs_2, jpi   ! vector opt.
            limitation_slop(zslpx(ji,jj,jk), zslpx(ji,jj,jk), zwx(ji-1,jj,jk), zwx(ji,jj,jk) )
            limitation_slop(zslpy(ji,jj,jk), zslpy(ji,jj,jk), zwy(ji,jj-1,jk), zwy(ji,jj,jk) )
         END DO
      END DO
   END DO
   !!-- MUSCL horizontal advective fluxes
   DO jk = 1, jpkm1
      DO jj = 2, jpjm1
         DO ji = fs_2, fs_jpim1   ! vector opt.
            vertical_adv_flux_i(zwx(ji,jj,jk), ji, zslpx(ji,jj,jk), zslpx(ji+1,jj,jk))
            vertical_adv_flux_j(zwy(ji,jj,jk), ji, zslpy(ji,jj,jk), zslpy(ji,jj+1,jk))
         END DO
      END DO
   END DO

   CALL mpp_lnk_3d(zwx, 1); CALL mpp_lnk_3d(zwy, 1)

   !-- Tracer advective trend
   DO jk = 1, jpkm1
      DO jj = 2, jpjm1
         DO ji = fs_2, fs_jpim1   ! vector opt.
            zbtr = 1. / ( e1e2t(ji,jj) * fse3t(ji,jj,jk) )
            ztra_i = zwx(ji-1,jj,jk) - zwx(ji,jj,jk)
            ztra_j = zwy(ji,jj-1,jk) - zwy(ji,jj,jk)
            pta(ji,jj,jk,jn) = pta(ji,jj,jk,jn) + zbtr * ( ztra_i + ztra_j )
         END DO
      END DO
   END DO
```

Prototype 1

```
CALL mpp_lnk_4d(ptb, 2)

DO jn = 1, kjpt             !==  loop over the tracers  ==!
   DO jk = 3, jpk-3
      DO jj = nldj, nlej
         DO ji = nldi, nlei
            !!! Initial slop !!!
            initial_slop_i(zzxm2, ji-2)
            initial_slop_i(zzxm1, ji-1)
            initial_slop_i(zzx, ji)
            initial_slop_i(zzxp1, ji+1)
            !!! Tracer slop & Limitation slop !!!
            tracer_slop(zzslpxm1, zzxm1, zzxm2)
            tracer_slop(zzslpx, zzx, zzxm1)
            tracer_slop(zzslpxp1, zzxp1, zzx)
            limitation_slop(zzslpxm1, zzslpxm1, zzxm2, zzxm1)
            limitation_slop(zzslpx, zzslpx, zzxm1, zzx)
            limitation_slop(zzslpxp1, zzslpxp1, zzx, zzxp1)
            !!! Horizontal advection flux (x-axes) !!!
            vertical_adv_flux_i(zzxf, ji, zzslpx, zzslpxp1)
            vertical_adv_flux_i(zzxfm1, ji-1, zzslpxm1, zzslpx)

            !!! Initial slop !!!
            initial_slop_j(zzym2, jj-2)
            initial_slop_j(zzym1, jj-1)
            initial_slop_j(zzy, jj)
            initial_slop_j(zzyp1, jj+1)
            !!! Tracer slop & Limitation slop !!!
            tracer_slop(zzslpym1, zzym1, zzym2)
            tracer_slop(zzslpy, zzy, zzym1)
            tracer_slop(zzslpyp1, zzyp1, zzy)
            limitation_slop(zzslpym1, zzslpym1, zzym2, zzym1)
            limitation_slop(zzslpy, zzslpy, zzym1, zzy)
            limitation_slop(zzslpyp1, zzslpyp1, zzy, zzyp1)
            !!! Horizontal advection flux (y-axes) !!!
            vertical_adv_flux_j(zzyf, jj, zzslpy, zzslpyp1)
            vertical_adv_flux_j(zzyfm1, jj-1, zzslpym1, zzslpy)

            !!! Initial slop !!!
            initial_slop_k(zzwzm1, jk-1)
            initial_slop_k(zzwz  , jk  )
            initial_slop_k(zzwzp1, jk+1)
            initial_slop_k(zzwzp2, jk+2)
            !!! Tracer slop & Limitation slop !!!
            tracer_slop(zzslpzm1, zzwzm1, zzwz)
            tracer_slop(zzslpz  , zzwz  , zzwzp1)
            tracer_slop(zzslpzp1, zzwzp1, zzwzp2)
            limitation_slop(zslpzzm1, zzslpzm1, zzwz  , zzwzm1)
            limitation_slop(zslpzz  , zzslpz  , zzwzp1, zzwz  )
            limitation_slop(zslpzzp1, zzslpzp1, zzwzp2, zzwzp1)
            !!! vertical advenction flux !!!
            vertical_adv_flux_k(zzwzf, jk, zslpzzm1, zslpzz)
            vertical_adv_flux_k(zzwzfp1, jk+1, zslpzz, zslpzzp1)

            ! add to the general tracer trends
            zbtr = 1. / ( e1e2t(ji,jj) * fse3t(ji,jj,jk) )
            ztra_i = zzxfm1 - zzxf
            ztra_j = zzyfm1 - zzyf
            ztra_k = zzwzfp1 - zzwzf
            pta(ji,jj,jk,jn) =  pta(ji,jj,jk,jn) + zbtr * ( ztra_i + ztra_j + ztra_k )
         END DO
      END DO
   END DO
END DO
```

Prototype 1

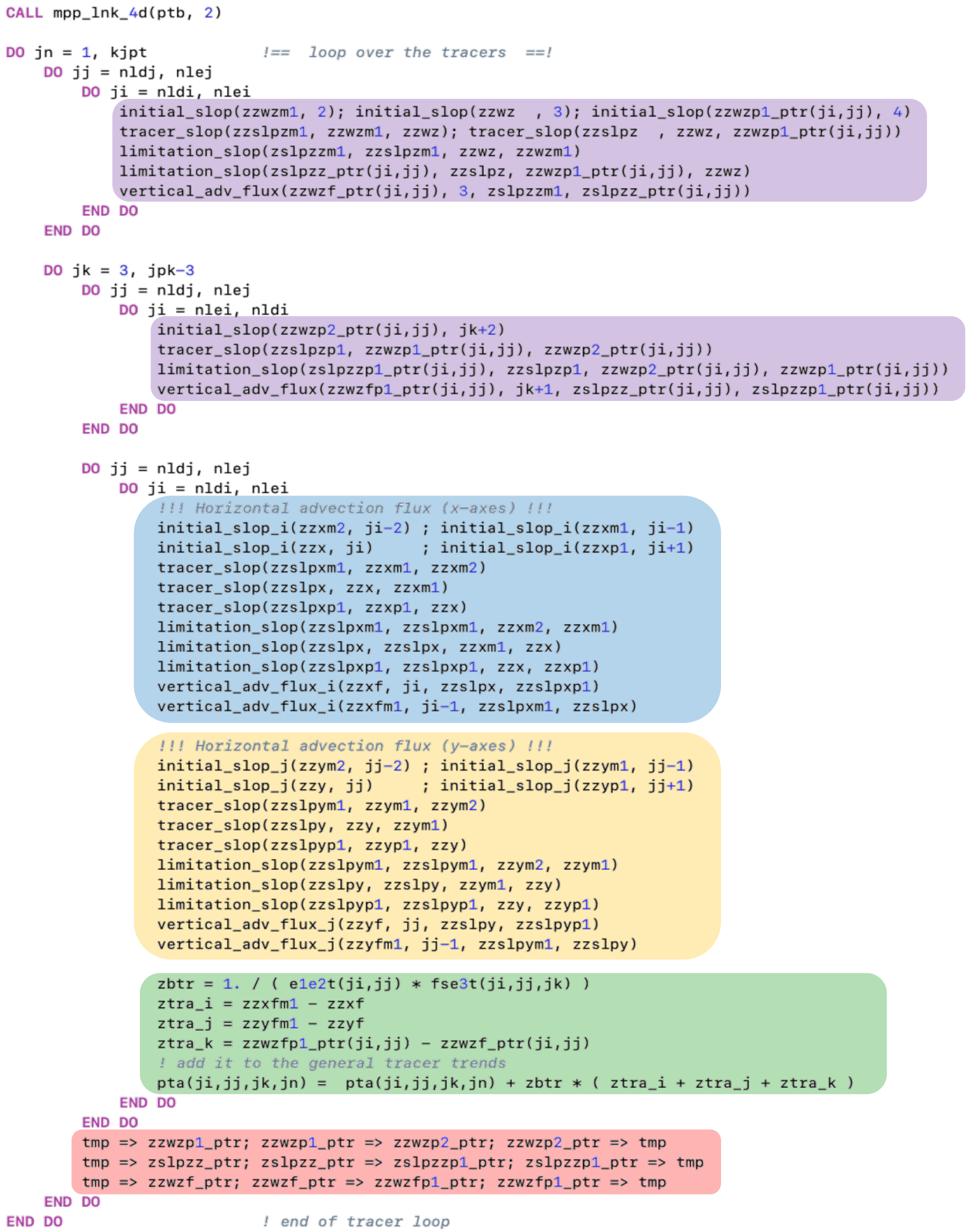
```
CALL mpp_lnk_4d(ptb, 2)

DO jn = 1, kjpt             !==  loop over the tracers  ==!
    DO jk = 3, jpk-3
        DO jj = nldj, nlej
            DO ji = nldi, nlei
                !!! Initial slop !!!
                initial_slop_i(zzxm2, ji-2)
                initial_slop_i(zzxm1, ji-1)
                initial_slop_i(zzx, ji)
                initial_slop_i(zzxp1, ji+1)
                !!! Tracer slop & Limitation slop !!!
                tracer_slop(zzslpxm1, zzxm1, zzxm2)
                tracer_slop(zzslpx, zzx, zzxm1)
                tracer_slop(zzslpxp1, zzxp1, zzx)
                limitation_slop(zzslpxm1, zzslpxm1, zzxm2, zzxm1)
                limitation_slop(zzslpx, zzslpx, zzxm1, zzx)
                limitation_slop(zzslpxp1, zzslpxp1, zzx, zzxp1)
                !!! Horizontal advection flux (x-axes) !!!
                vertical_adv_flux_i(zzxf, ji, zzslpx, zzslpxp1)
                vertical_adv_flux_i(zzxfm1, ji-1, zzslpxm1, zzslpx)

                !!! Initial slop !!!
                initial_slop_j(zzym2, jj-2)
                initial_slop_j(zzym1, jj-1)
                initial_slop_j(zzy, jj)
                initial_slop_j(zzyp1, jj+1)
                !!! Tracer slop & Limitation slop !!!
                tracer_slop(zzslpym1, zzym1, zzym2)
                tracer_slop(zzslpy, zzy, zzym1)
                tracer_slop(zzslpyp1, zzyp1, zzy)
                limitation_slop(zzslpym1, zzslpym1, zzym2, zzym1)
                limitation_slop(zzslpy, zzslpy, zzym1, zzy)
                limitation_slop(zzslpyp1, zzslpyp1, zzy, zzyp1)
                !!! Horizontal advection flux (y-axes) !!!
                vertical_adv_flux_j(zzyf, jj, zzslpy, zzslpyp1)
                vertical_adv_flux_j(zzyfm1, jj-1, zzslpym1, zzslpy)

                !!! Initial slop !!!
                initial_slop_k(zzwzm1, jk-1)
                initial_slop_k(zzwz  , jk  )
                initial_slop_k(zzwzp1, jk+1)
                initial_slop_k(zzwzp2, jk+2)
                !!! Tracer slop & Limitation slop !!!
                tracer_slop(zzslpzm1, zzwzm1, zzwz)
                tracer_slop(zzslpz  , zzwz  , zzwzp1)
                tracer_slop(zzslpzp1, zzwzp1, zzwzp2)
                limitation_slop(zslpzzm1, zzslpzm1, zzwz  , zzwzm1)
                limitation_slop(zslpzz  , zzslpz  , zzwzp1, zzwz  )
                limitation_slop(zslpzzp1, zzslpzp1, zzwzp2, zzwzp1)
                !!! vertical advenction flux !!!
                vertical_adv_flux_k(zzwzf, jk, zslpzzm1, zslpzz)
                vertical_adv_flux_k(zzwzfp1, jk+1, zslpzz, zslpzzp1)

                ! add to the general tracer trends
                zbtr = 1. / ( e1e2t(ji,jj) * fse3t(ji,jj,jk) )
                ztra_i = zzxfm1 - zzxf
                ztra_j = zzyfm1 - zzyf
                ztra_k = zzwzfp1 - zzwzf
                pta(ji,jj,jk,jn) =  pta(ji,jj,jk,jn) + zbtr * ( ztra_i + ztra_j + ztra_k )
            END DO
        END DO
    END DO
END DO
```

Prototype 2

```
CALL mpp_lnk_4d(ptb, 2)

DO jn = 1, kjpt              !==  loop over the tracers  ==!
    DO jj = nldj, nlej
        DO ji = nldi, nlei
            initial_slop(zzwzm1, 2); initial_slop(zzwz  , 3); initial_slop(zzwzp1_ptr(ji,jj), 4)
            tracer_slop(zzslpzm1, zzwzm1, zzwz); tracer_slop(zzslpz  , zzwz, zzwzp1_ptr(ji,jj))
            limitation_slop(zslpzzm1, zzslpzm1, zzwz, zzwzm1)
            limitation_slop(zslpzz_ptr(ji,jj), zzslpz, zzwzp1_ptr(ji,jj), zzwz)
            vertical_adv_flux(zzwzf_ptr(ji,jj), 3, zslpzzm1, zslpzz_ptr(ji,jj))
        END DO
    END DO

    DO jk = 3, jpk-3
        DO jj = nldj, nlej
            DO ji = nlei, nldi
                initial_slop(zzwzp2_ptr(ji,jj), jk+2)
                tracer_slop(zzslpzp1, zzwzp1_ptr(ji,jj), zzwzp2_ptr(ji,jj))
                limitation_slop(zslpzzp1_ptr(ji,jj), zzslpzp1, zzwzp2_ptr(ji,jj), zzwzp1_ptr(ji,jj))
                vertical_adv_flux(zzwzfp1_ptr(ji,jj), jk+1, zslpzz_ptr(ji,jj), zslpzzp1_ptr(ji,jj))
            END DO
        END DO

        DO jj = nldj, nlej
            DO ji = nldi, nlei
                !!! Horizontal advection flux (x-axes) !!!
                initial_slop_i(zzxm2, ji-2) ; initial_slop_i(zzxm1, ji-1)
                initial_slop_i(zzx, ji)     ; initial_slop_i(zzxp1, ji+1)
                tracer_slop(zzslpxm1, zzxm1, zzxm2)
                tracer_slop(zzslpx, zzx, zzxm1)
                tracer_slop(zzslpxp1, zzxp1, zzx)
                limitation_slop(zzslpxm1, zzslpxm1, zzxm2, zzxm1)
                limitation_slop(zzslpx, zzslpx, zzxm1, zzx)
                limitation_slop(zzslpxp1, zzslpxp1, zzx, zzxp1)
                vertical_adv_flux_i(zzxf, ji, zzslpx, zzslpxp1)
                vertical_adv_flux_i(zzxfm1, ji-1, zzslpxm1, zzslpx)

                !!! Horizontal advection flux (y-axes) !!!
                initial_slop_j(zzym2, jj-2) ; initial_slop_j(zzym1, jj-1)
                initial_slop_j(zzy, jj)     ; initial_slop_j(zzyp1, jj+1)
                tracer_slop(zzslpym1, zzym1, zzym2)
                tracer_slop(zzslpy, zzy, zzym1)
                tracer_slop(zzslpyp1, zzyp1, zzy)
                limitation_slop(zzslpym1, zzslpym1, zzym2, zzym1)
                limitation_slop(zzslpy, zzslpy, zzym1, zzy)
                limitation_slop(zzslpyp1, zzslpyp1, zzy, zzyp1)
                vertical_adv_flux_j(zzyf, jj, zzslpy, zzslpyp1)
                vertical_adv_flux_j(zzyfm1, jj-1, zzslpym1, zzslpy)

                zbtr = 1. / ( e1e2t(ji,jj) * fse3t(ji,jj,jk) )
                ztra_i = zzxfm1 - zzxf
                ztra_j = zzyfm1 - zzyf
                ztra_k = zzwzfp1_ptr(ji,jj) - zzwzf_ptr(ji,jj)
                ! add it to the general tracer trends
                pta(ji,jj,jk,jn) =  pta(ji,jj,jk,jn) + zbtr * ( ztra_i + ztra_j + ztra_k )
            END DO
        END DO
        tmp => zzwzp1_ptr; zzwzp1_ptr => zzwzp2_ptr; zzwzp2_ptr => tmp
        tmp => zslpzz_ptr; zslpzz_ptr => zslpzzp1_ptr; zslpzzp1_ptr => tmp
        tmp => zzwzf_ptr; zzwzf_ptr => zzwzfp1_ptr; zzwzfp1_ptr => tmp
    END DO
END DO                      ! end of tracer loop
```

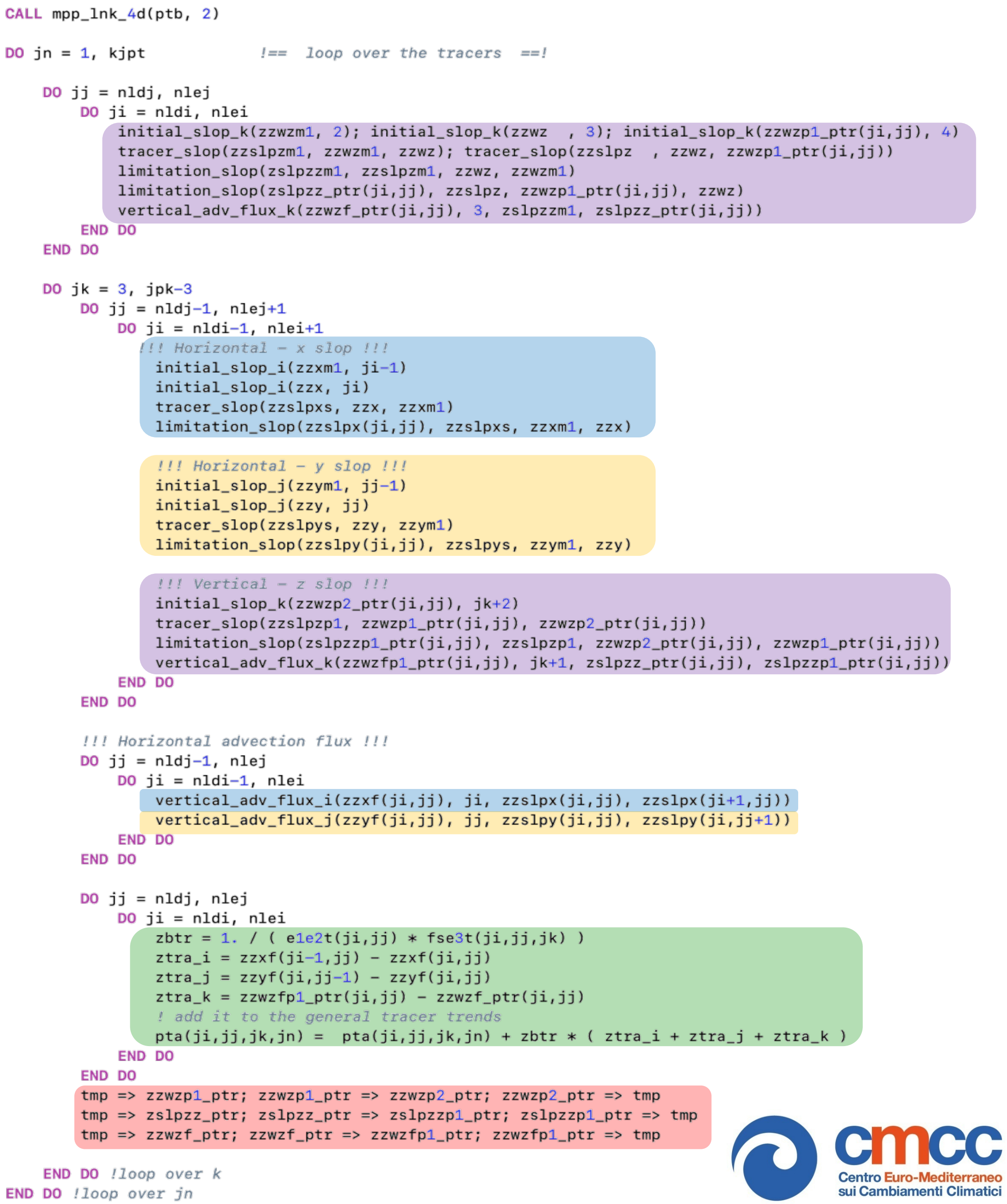
Centro Euro-Mediterraneo sui Cambiamenti Climatici

Prototype 2



```
CALL mpp_lnk_4d(ptb, 2)

DO jn = 1, kjpt            !==  loop over the tracers  ==!
    DO jj = nldj, nlej
        DO ji = nldi, nlei
            initial_slop(zzwzm1, 2); initial_slop(zzwz  , 3); initial_slop(zzwzp1_ptr(ji,jj), 4)
            tracer_slop(zzslpzm1, zzwzm1, zzwz); tracer_slop(zzslpz  , zzwz, zzwzp1_ptr(ji,jj))
            limitation_slop(zslpzzm1, zzslpzm1, zzwz, zzwzm1)
            limitation_slop(zslpzz_ptr(ji,jj), zzslpz, zzwzp1_ptr(ji,jj), zzwz)
            vertical_adv_flux(zzwzf_ptr(ji,jj), 3, zslpzzm1, zslpzz_ptr(ji,jj))
        END DO
    END DO

    DO jk = 3, jpk-3
        DO jj = nldj, nlej
            DO ji = nlei, nldi
                initial_slop(zzwzp2_ptr(ji,jj), jk+2)
                tracer_slop(zzslpzp1, zzwzp1_ptr(ji,jj), zzwzp2_ptr(ji,jj))
                limitation_slop(zslpzzp1_ptr(ji,jj), zzslpzp1, zzwzp2_ptr(ji,jj), zzwzp1_ptr(ji,jj))
                vertical_adv_flux(zzwzfp1_ptr(ji,jj), jk+1, zslpzz_ptr(ji,jj), zslpzzp1_ptr(ji,jj))
            END DO
        END DO

        DO jj = nldj, nlej
            DO ji = nldi, nlei
                !!! Horizontal advection flux (x-axes) !!!
                initial_slop_i(zzxm2, ji-2) ; initial_slop_i(zzxm1, ji-1)
                initial_slop_i(zzx, ji)     ; initial_slop_i(zzxp1, ji+1)
                tracer_slop(zzslpxm1, zzxm1, zzxm2)
                tracer_slop(zzslpx, zzx, zzxm1)
                tracer_slop(zzslpxp1, zzxp1, zzx)
                limitation_slop(zzslpxm1, zzslpxm1, zzxm2, zzxm1)
                limitation_slop(zzslpx, zzslpx, zzxm1, zzx)
                limitation_slop(zzslpxp1, zzslpxp1, zzx, zzxp1)
                vertical_adv_flux_i(zzxf, ji, zzslpx, zzslpxp1)
                vertical_adv_flux_i(zzxfm1, ji-1, zzslpxm1, zzslpx)

                !!! Horizontal advection flux (y-axes) !!!
                initial_slop_j(zzym2, jj-2) ; initial_slop_j(zzym1, jj-1)
                initial_slop_j(zzy, jj)     ; initial_slop_j(zzyp1, jj+1)
                tracer_slop(zzslpym1, zzym1, zzym2)
                tracer_slop(zzslpy, zzy, zzym1)
                tracer_slop(zzslpyp1, zzyp1, zzy)
                limitation_slop(zzslpym1, zzslpym1, zzym2, zzym1)
                limitation_slop(zzslpy, zzslpy, zzym1, zzy)
                limitation_slop(zzslpyp1, zzslpyp1, zzy, zzyp1)
                vertical_adv_flux_j(zzyf, jj, zzslpy, zzslpyp1)
                vertical_adv_flux_j(zzyfm1, jj-1, zzslpym1, zzslpy)

                zbtr = 1. / ( e1e2t(ji,jj) * fse3t(ji,jj,jk) )
                ztra_i = zzxfm1 - zzxf
                ztra_j = zzyfm1 - zzyf
                ztra_k = zzwzfp1_ptr(ji,jj) - zzwzf_ptr(ji,jj)
                ! add it to the general tracer trends
                pta(ji,jj,jk,jn) =  pta(ji,jj,jk,jn) + zbtr * ( ztra_i + ztra_j + ztra_k )
            END DO
        END DO
        tmp => zzwzp1_ptr; zzwzp1_ptr => zzwzp2_ptr; zzwzp2_ptr => tmp
        tmp => zslpzz_ptr; zslpzz_ptr => zslpzzp1_ptr; zslpzzp1_ptr => tmp
        tmp => zzwzf_ptr; zzwzf_ptr => zzwzfp1_ptr; zzwzfp1_ptr => tmp
    END DO
END DO                     ! end of tracer loop
```

Prototype 3

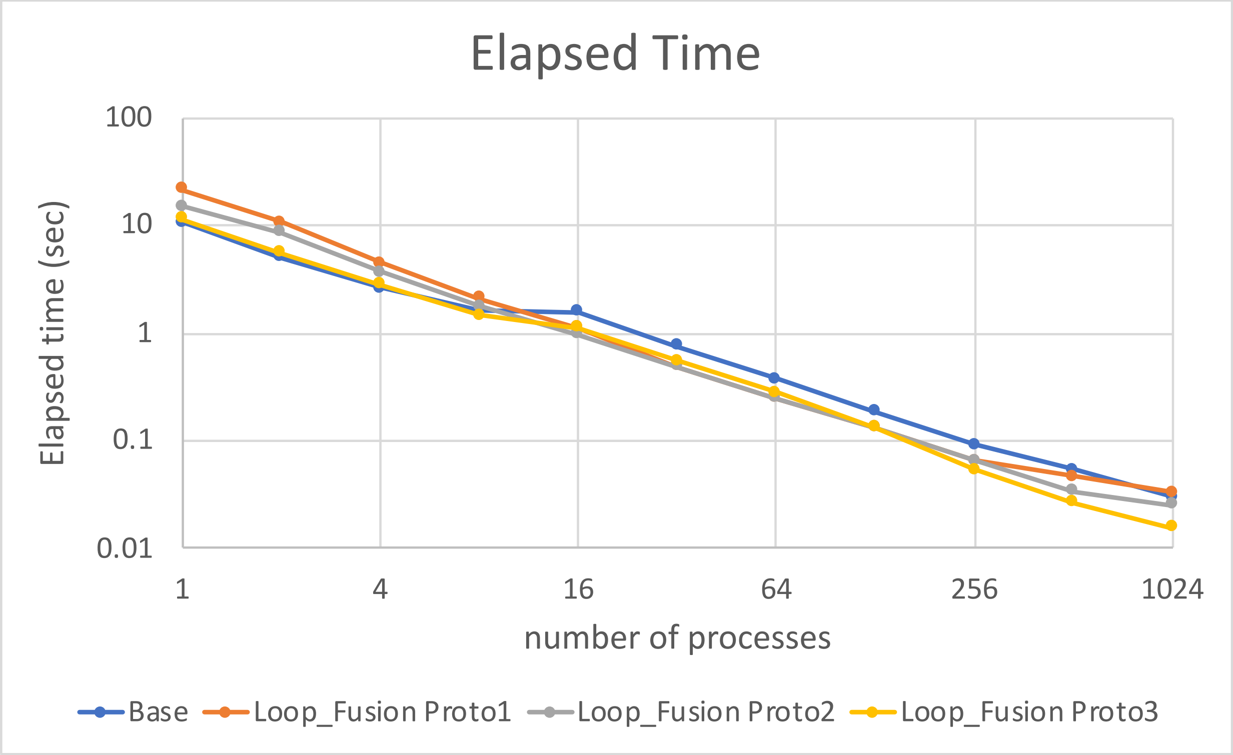


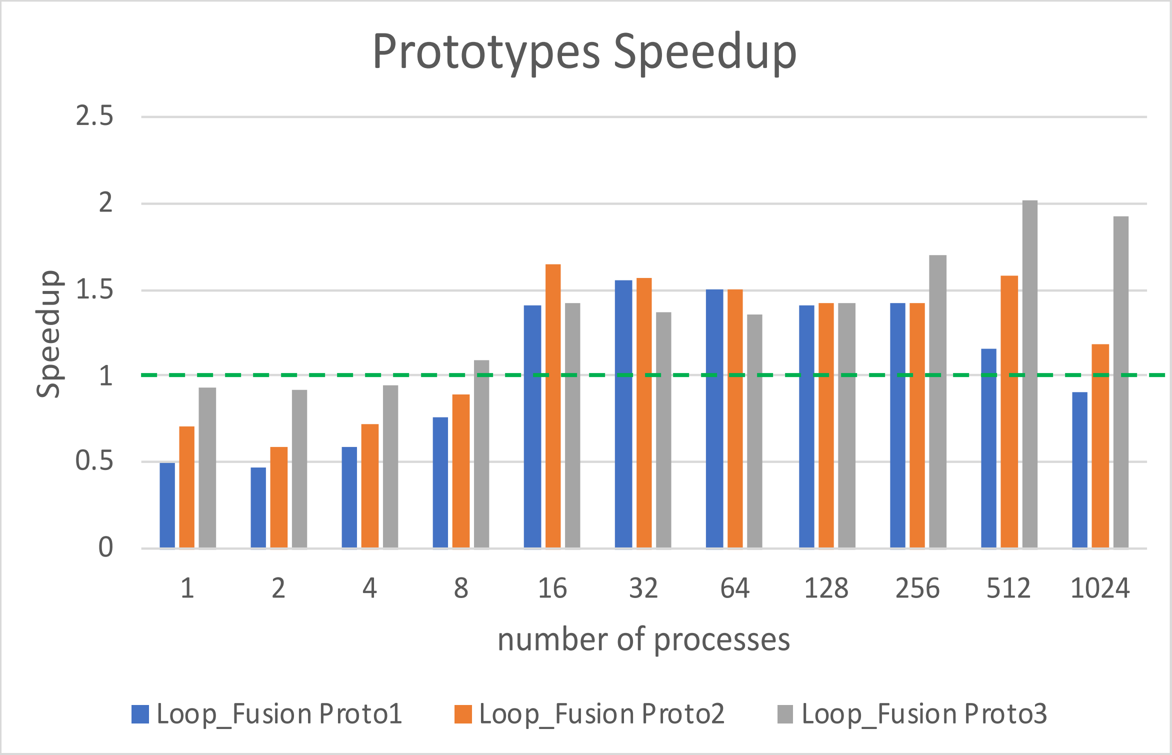
```
CALL mpp_lnk_4d(ptb, 2)

DO jn = 1, kjpt            !==  loop over the tracers  ==!

    DO jj = nldj, nlej
        DO ji = nldi, nlei
            initial_slop_k(zzwzm1, 2); initial_slop_k(zzwz  , 3); initial_slop_k(zzwzp1_ptr(ji,jj), 4)
            tracer_slop(zzslpzm1, zzwzm1, zzwz); tracer_slop(zzslpz  , zzwz, zzwzp1_ptr(ji,jj))
            limitation_slop(zslpzzm1, zzslpzm1, zzwz, zzwzm1)
            limitation_slop(zslpzz_ptr(ji,jj), zzslpz, zzwzp1_ptr(ji,jj), zzwz)
            vertical_adv_flux_k(zzwzf_ptr(ji,jj), 3, zslpzzm1, zslpzz_ptr(ji,jj))
        END DO
    END DO

    DO jk = 3, jpk-3
        DO jj = nldj-1, nlej+1
            DO ji = nldi-1, nlei+1
               !!! Horizontal - x slop !!!
                initial_slop_i(zzxm1, ji-1)
                initial_slop_i(zzx, ji)
                tracer_slop(zzslpxs, zzx, zzxm1)
                limitation_slop(zzslpx(ji,jj), zzslpxs, zzxm1, zzx)

                !!! Horizontal - y slop !!!
                initial_slop_j(zzym1, jj-1)
                initial_slop_j(zzy, jj)
                tracer_slop(zzslpys, zzy, zzym1)
                limitation_slop(zzslpy(ji,jj), zzslpys, zzym1, zzy)

                !!! Vertical - z slop !!!
                initial_slop_k(zzwzp2_ptr(ji,jj), jk+2)
                tracer_slop(zzslpzp1, zzwzp1_ptr(ji,jj), zzwzp2_ptr(ji,jj))
                limitation_slop(zslpzzp1_ptr(ji,jj), zzslpzp1, zzwzp2_ptr(ji,jj), zzwzp1_ptr(ji,jj))
                vertical_adv_flux_k(zzwzfp1_ptr(ji,jj), jk+1, zslpzz_ptr(ji,jj), zslpzzp1_ptr(ji,jj))
            END DO
        END DO

        !!! Horizontal advection flux !!!
        DO jj = nldj-1, nlej
            DO ji = nldi-1, nlei
                vertical_adv_flux_i(zzxf(ji,jj), ji, zzslpx(ji,jj), zzslpx(ji+1,jj))
                vertical_adv_flux_j(zzyf(ji,jj), jj, zzslpy(ji,jj), zzslpy(ji,jj+1))
            END DO
        END DO

        DO jj = nldj, nlej
            DO ji = nldi, nlei
                zbtr = 1. / ( e1e2t(ji,jj) * fse3t(ji,jj,jk) )
                ztra_i = zzxf(ji-1,jj) - zzxf(ji,jj)
                ztra_j = zzyf(ji,jj-1) - zzyf(ji,jj)
                ztra_k = zzwzfp1_ptr(ji,jj) - zzwzf_ptr(ji,jj)
                ! add it to the general tracer trends
                pta(ji,jj,jk,jn) =  pta(ji,jj,jk,jn) + zbtr * ( ztra_i + ztra_j + ztra_k )
            END DO
        END DO
        tmp => zzwzp1_ptr; zzwzp1_ptr => zzwzp2_ptr; zzwzp2_ptr => tmp
        tmp => zslpzz_ptr; zslpzz_ptr => zslpzzp1_ptr; zslpzzp1_ptr => tmp
        tmp => zzwzf_ptr; zzwzf_ptr => zzwzfp1_ptr; zzwzfp1_ptr => tmp

    END DO !loop over k
END DO !loop over jn
```

cmcc
Centro Euro-Mediterraneo sui Cambiamenti Climatici

Preliminary Performance Analysis

- Test executed on Intel Xeon based Architecture
  - 16 cores per node
- Global domain: 2240 x 1500 x 31 points
- Smallest sub-domain (with 1024 cores): 74 x 51 x 31 points

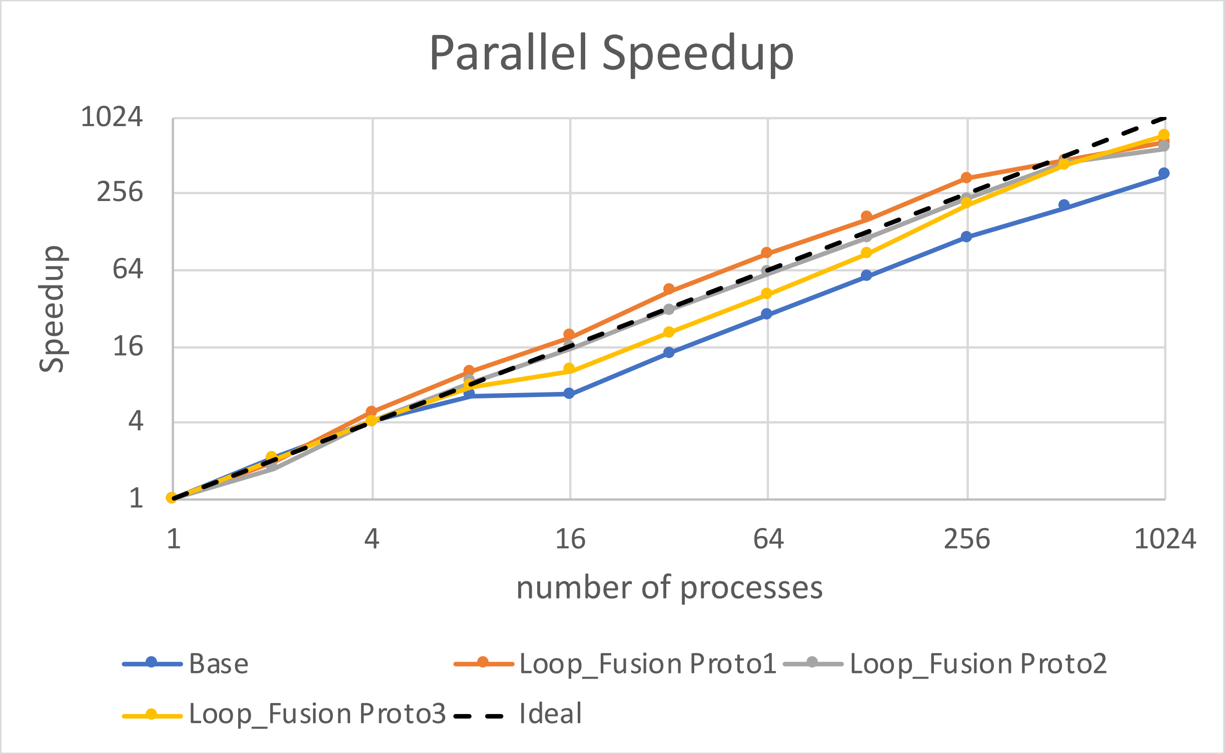


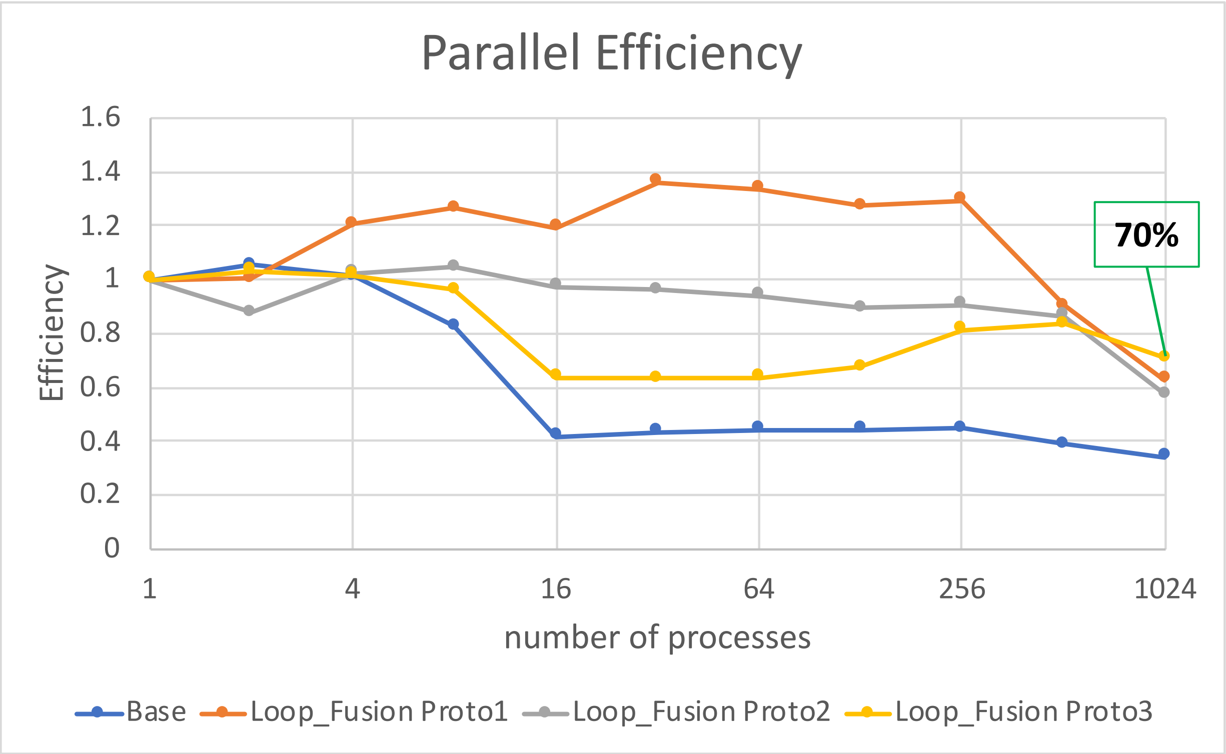


Elapsed time of the baseline over the elapsed time of the prototype: $S=T_{base}/T_{proto}$

Preliminary Performance Analysis

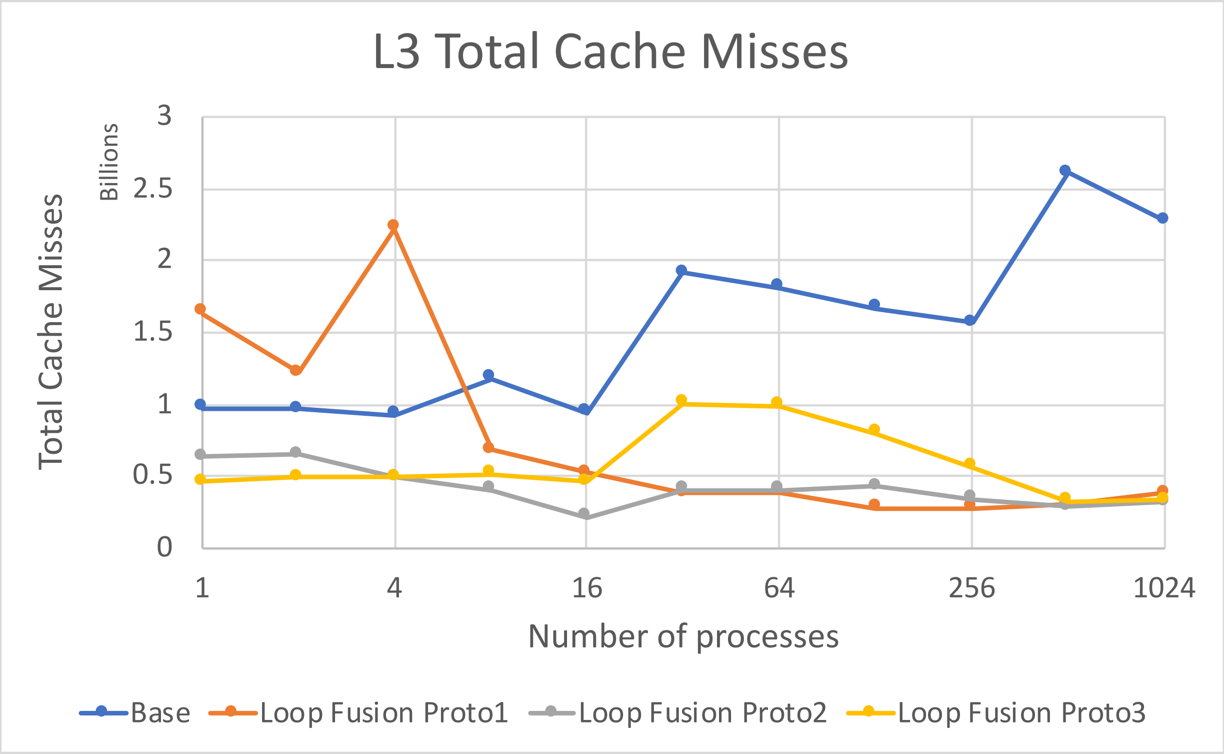
- Global domain: 2240 x 1500 x 31 points
- Smallest sub-domain (with 1024 cores): 74 x 51 x 31 points

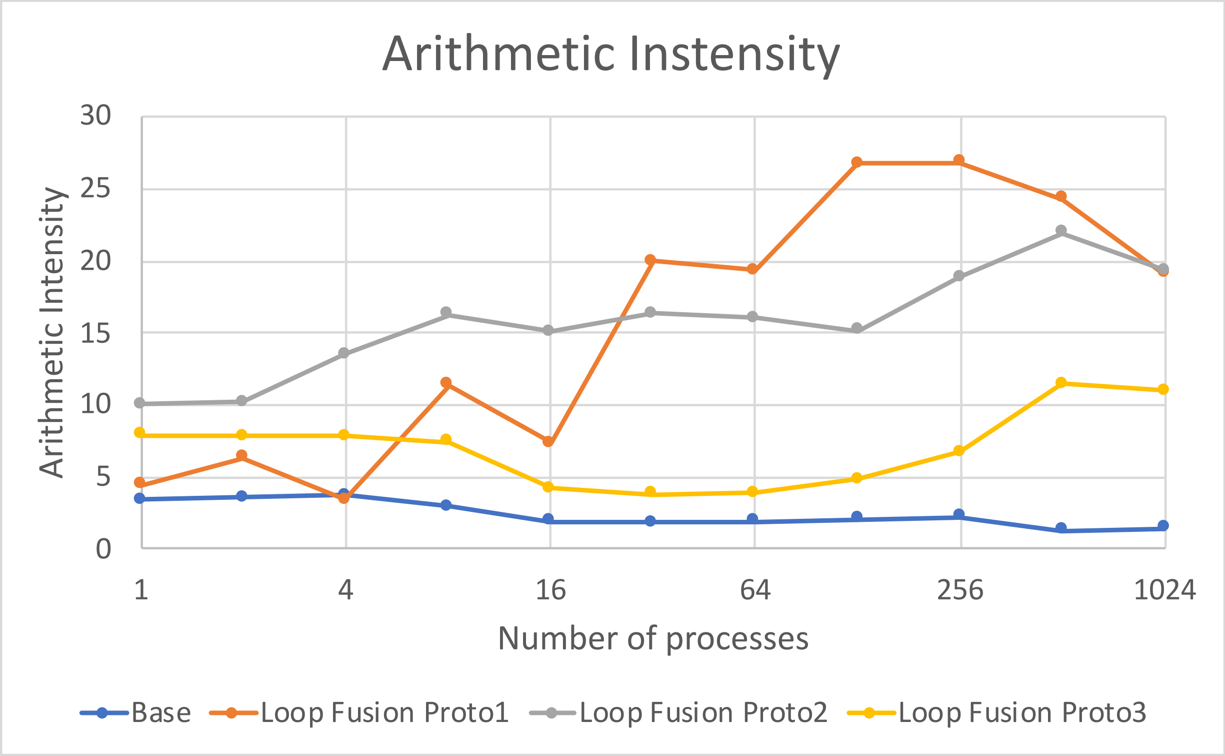




Preliminary Performance Analysis

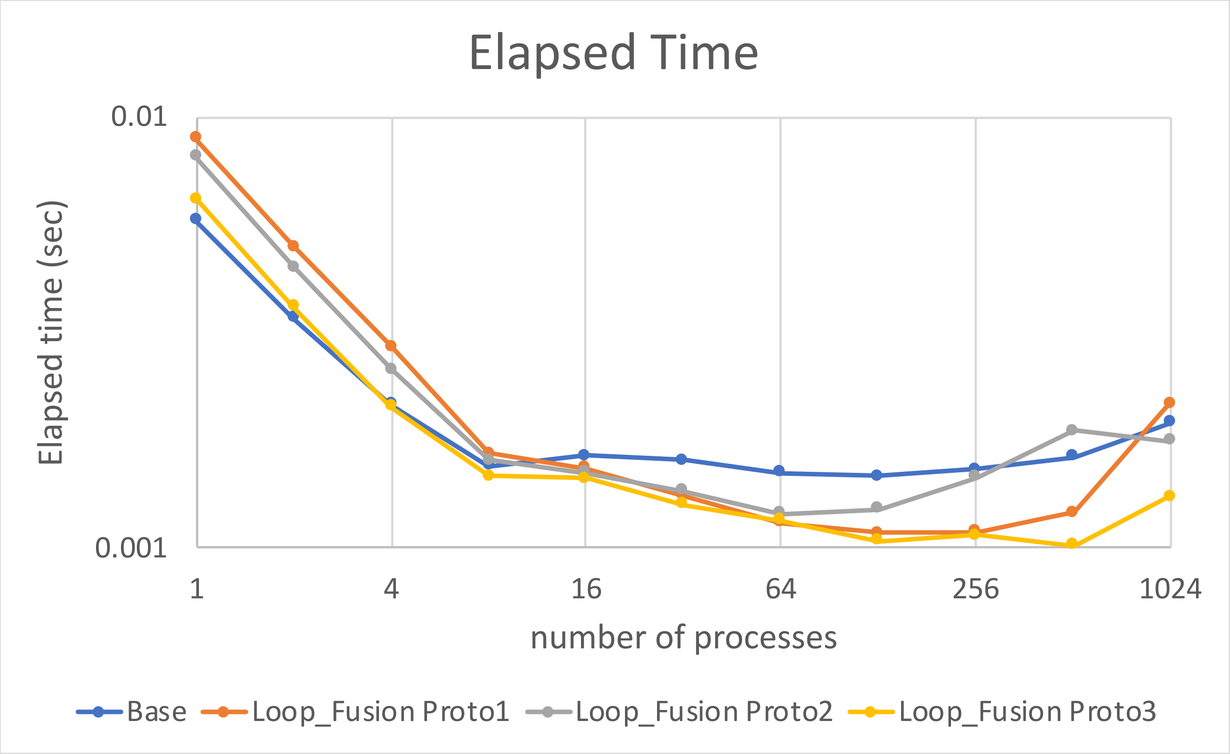
- Global domain: 2240 x 1500 x 31 points
- Smallest sub-domain (with 1024 cores): 74 x 51 x 31 points

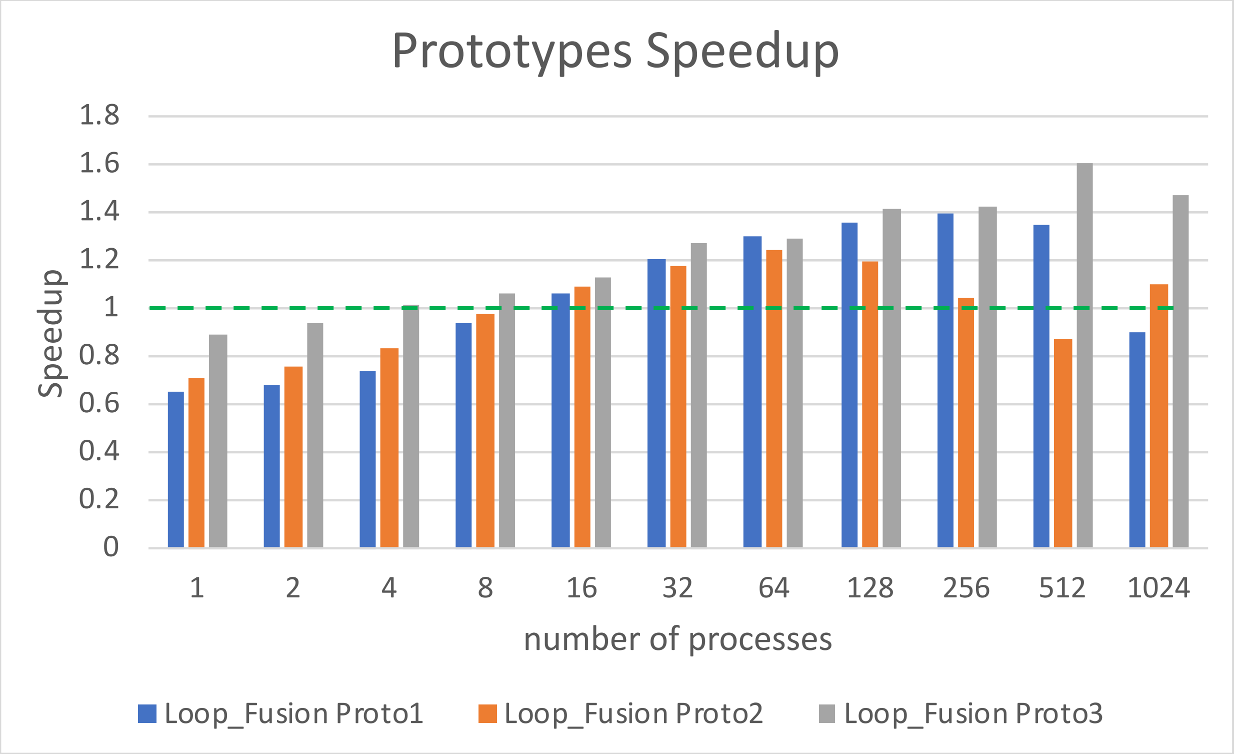




Preliminary Performance Analysis

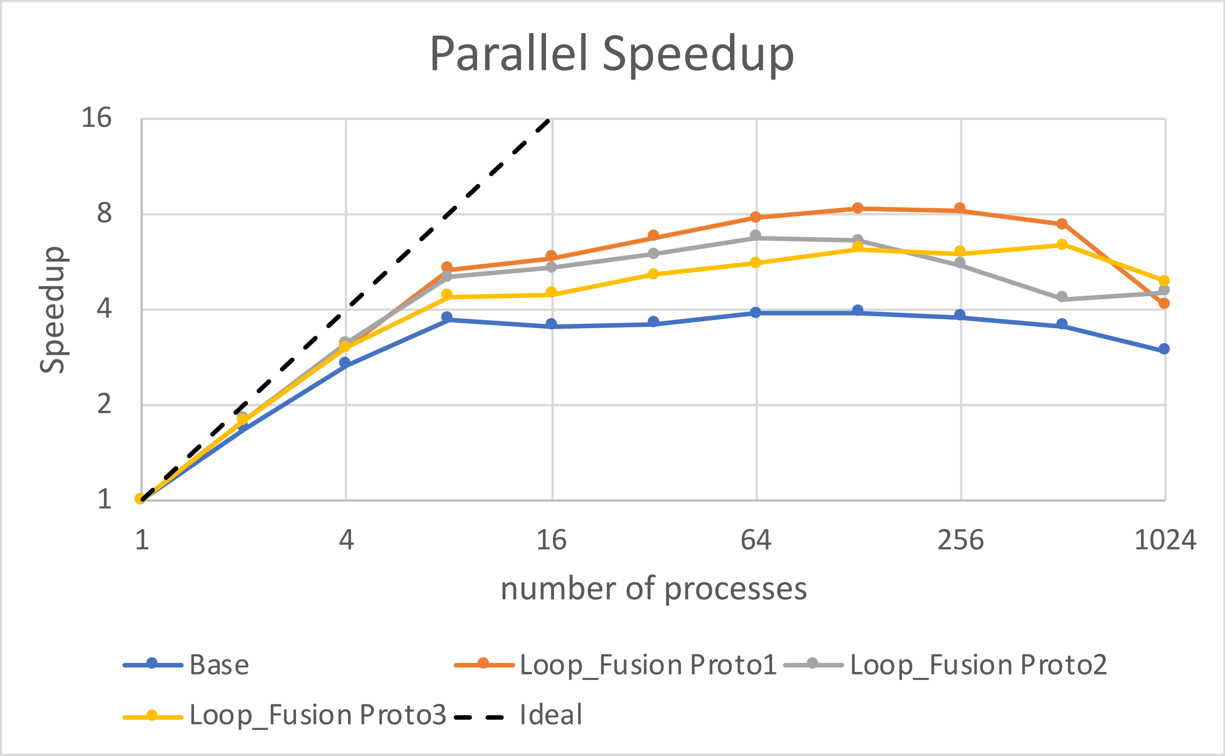
- Global domain: 70 x 46 x 19 points
- Sub-domain with 64 cores: 13 x 10 x 19 points

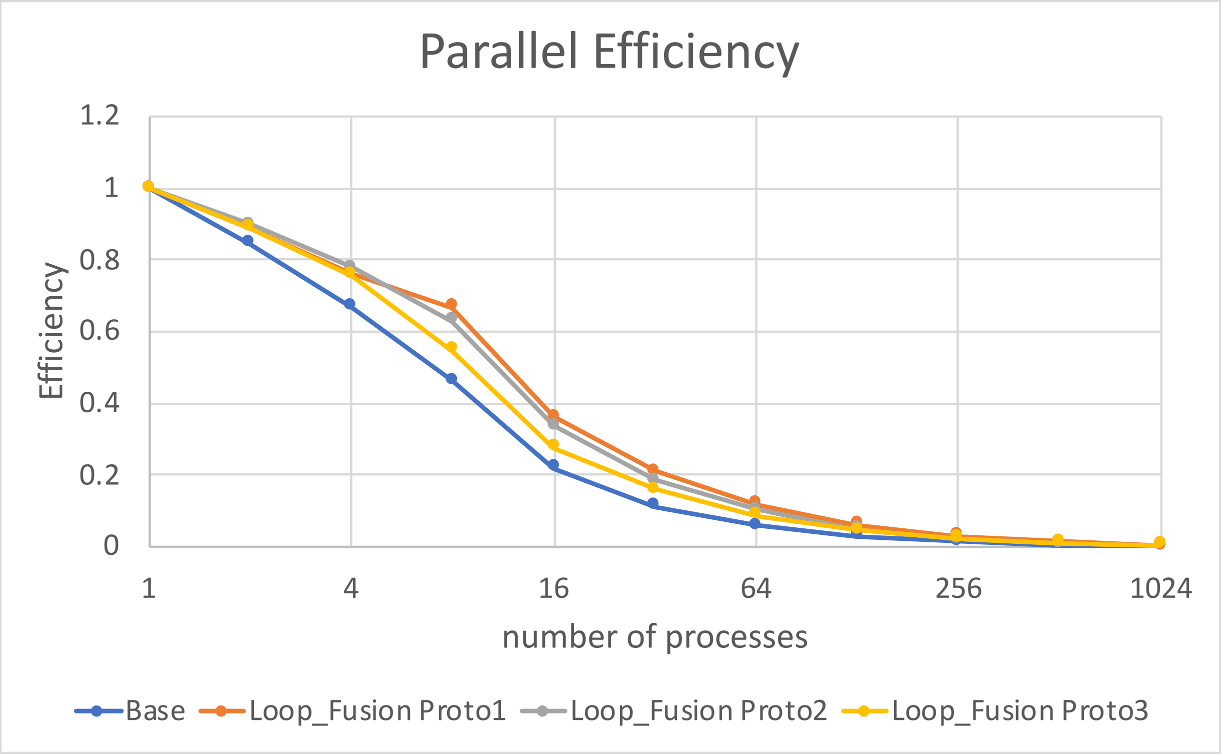




Preliminary Performance Analysis

- Global domain: 70 x 46 x 19 points
- Sub-domain with 64 cores: 13 x 10 x 19 points


Parallel Speedup
Speedup
number of processes
Base
Loop_Fusion Proto1
Loop_Fusion Proto2
Loop_Fusion Proto3
Ideal


Parallel Efficiency
Efficiency
number of processes
Base
Loop_Fusion Proto1
Loop_Fusion Proto2
Loop_Fusion Proto3