Schweizerische Eidgenossenschaft
Confédération suisse
Confederazione Svizzera
Confederaziun svizra

Update of STELLA developments

Carlos Osuna, C2SM (ETH)
Oliver Fuhrer, MeteoSwiss

STELLA Reminder (1/2)

```
  DO k = 1, ke
!CDIR OUTERUNROLL=4
    DO   j  = 2 , je-1
!CDIR ON_ADB(lap)
!CDIR ON_ADB(s)
      DO i  = 2 , ie-1
        lap (i,j,k)   = s (i+1,j,k) + s (i-1,j,k) - 2.0_ireals*s(i,j,k) &
                      +  crlato(j)*(s(i,j+1,k) - s(i,j  ,k))           &
                      -  crlatu(j)*(s(i,j  ,k) - s(i,j-1,k))
      ENDDO
    ENDDO
  ENDDO
```

- Abstract…
  - explicit data structure (i,j,k)
  - explicit loops and loop order
  - parallelization (e.g. NEc, OpenMP, CUDA, …)
  - optimizations (blocking, fusion, …)

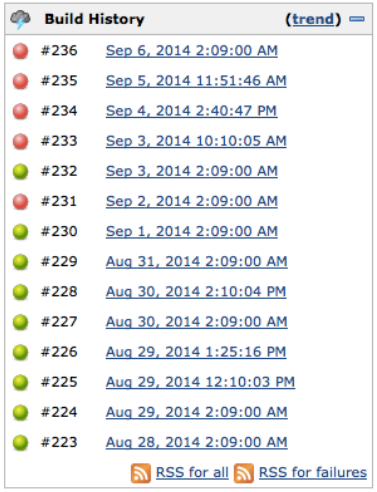
STELLA Reminder (2/2)

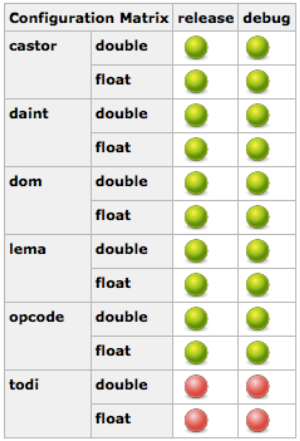
```
__ACC__
static T Do(Context ctx)
{
    ctx[data_out::Center()] = - (T)2.0 * ctx[data_in::Center()]
      + ctx[data_in::At(iplus1)] + ctx[data_in::At(iminus1)]
      + ctx[crlatvo::Center()] * ctx[Call<Delta>::With(jplus1 , data_in::Center())]
      + ctx[crlatvu::Center()] * ctx[Call<Delta>::With(jminus1, data_in::Center())];
}
```

```
// setup the tracer stencil
StencilCompiler::Build(
    stencil_,
    "HorizontalDiffusionTracers",
    dycoreRepository.calculationDomain(),
    StencilConfiguration<Real, HorizontalDiffusionTracersBlockSize>(),
    pack_parameters(
        Param<data_out, cInOut>(data_out_),
        Param<data_in, cIn>(data_in_),
    ),
    concatenate_sweeps(
        define_sweep<cKIncrement>(
            define_stages(
                StencilStage<LapStage, IJRange<cComplete,-2,2,-2,2>,
                                       KRange<FullDomain,0,0> >(),
            )
        )
    )
);
```

Code quality

- 35’000 lines of code (shared, stencil, comm, verif, serial, …)
- 20’000 lines of unit-tests
- Daily builds and regression tests





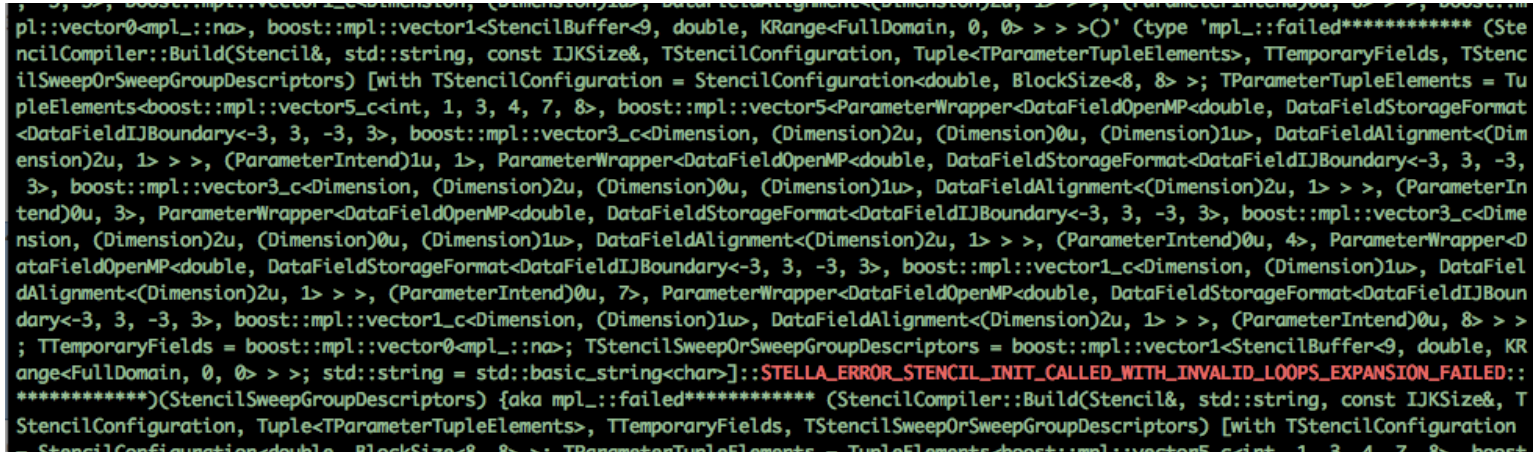
- No error caught by user to date!

Code owner

- Tobias Gysi (CS, SCS) left the project
- Ben Cumming (Math, CSCS) code owner since 08.2013
  - Minor developments
  - Code reviews
- Carlos Osuna (Physics, C2SM) main developer since 04.2014
  - Major developments

Compile time errors

- Nice example



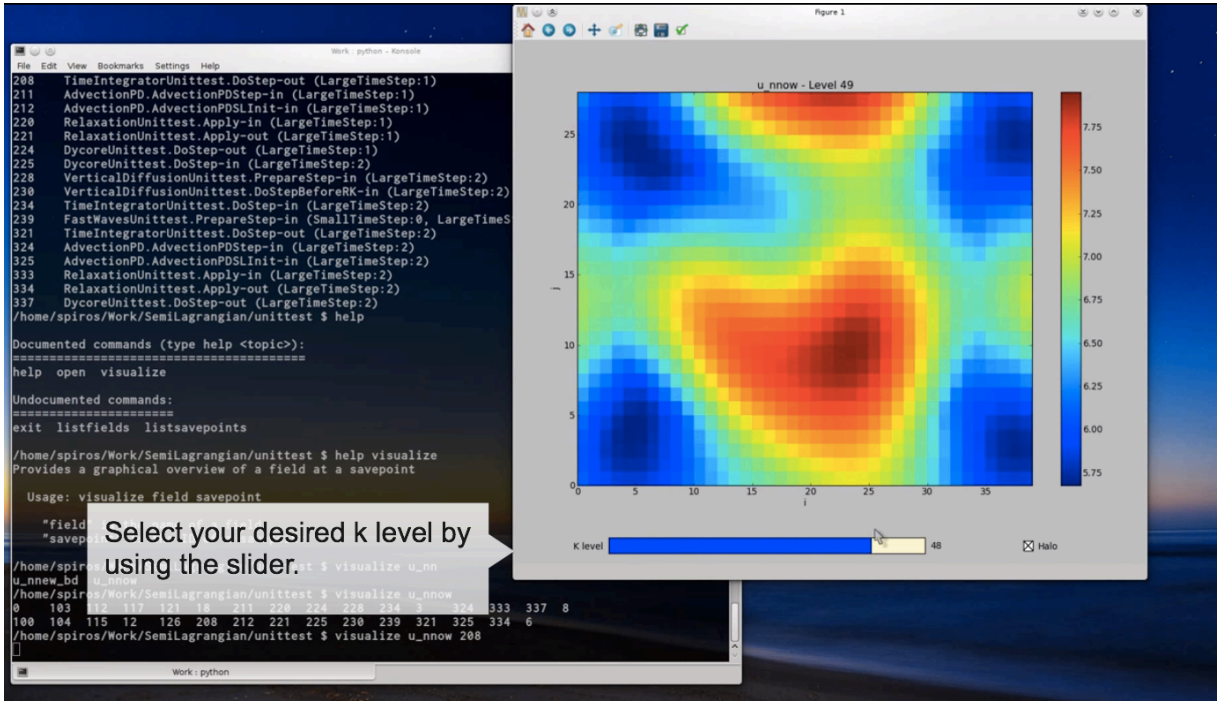
- All syntax protections of STELLA now produce a common error message

  `STELLA_ERROR_STENCIL_INIT_CALLED_WITH_INVALID_LOOPS_EXPANSION_FAILED`

- Just “grep STELLA_ERROR” compiler output
- Documentation with error message explanations in progress

Debug / verbose mode

- Fields and temporaries (automatically) initialized with NaN
- C++, Fortran and (prototype) Python interface to serialized fields



Debug / verbose mode

- Logging (verbose) mode (STELLA_ENABLE_LOGGING)

```
Stencil Coriolis Apply() started...
Bounding Box Info:
utens                          RW  from 0 0 0 to  41  26  59
vtens                          RW  from 0 0 0 to  41  26  59
u_nnow                         R   from -1  0 0 to  41  27  59
v_nnow                         R   from 0 -1  0 to  42  26  59
fc                             R   from -1  -1  0 to  41  26  59
Stencil Coriolis Apply() end
Stencil ConvertTemperatureTP Apply(const IJBoundary&) started...
Bounding Box Info:
tp_nnow                        RW  from -3  -3  0 to  44  29  59
tp_nnew_bd                     RW  from -3  -3  0 to  44  29  59
t_nnow                         R   from -3  -3  0 to  44  29  59
t_nnew_bd                      R   from -3  -3  0 to  44  29  59
t0                             R   from -3  -3  0 to  44  29  59
Stencil ConvertTemperatureTP Apply(const IJBoundary&) end
Stencil VerticalDiffusionPrepareStep Apply() started...
Bounding Box Info:
vdtch                          RW  from 0 0 59  to  42  27  59
vdtcm                          RW  from 0 0 59  to  42  27  59
kh                             RW  from 0 0 1 to  41  26  59
tmkvm                          RW  from -1  -1  1 to  42  27  59
tmkvw                          RW  from 0 0 1 to  41  26  59
sqrtgrhors                     RW  from 0 0 0 to  41  26  59
u_nnow                         R   from -1  0 59  to  42  27  59
v_nnow                         R   from 0 -1  59  to  42  27  59
rho                            R   from -1  -1  0 to  42  27  59
sqrtgrs                        R   from 0 0 0 to  41  26  59
sqrtgrw                        R   from -1  -1  1 to  42  27  59
tkvh                           R   from -1  -1  0 to  42  27  58
tkvm                           R   from -1  -1  0 to  42  27  58
p_s                            R   from 0 0 59  to  42  27  59
t_s                            R   from 0 0 59  to  42  27  59
qv_s                           R   from 0 0 59  to  42  27  59
tch                            R   from 0 0 59  to  42  27  59
```

Currently
- Workflow
- Inputs / Outputs
- Dependencies
- Halo-updates

Outlook
- Stage level info
- DAG
- … (your ideas)

Static parser

- Static parser for STELLA code being developed (PhD Tobias Gysi)
- Only possible due to DSL syntax
- Example

```
-> stencil: AdvectionPDBottPrepareTracers1
-> stage PremultiplyWithRhoStage1
-> method FullDomain
-> input fields
   -> data1_nnow: (0, 0, 0)
   -> rho: (0, 0, 0)
-> output fields
   -> data1: (0, 0, 0)
-> stage PremultiplyWithRhoStage2
-> method FullDomain
-> input fields
   -> data1_nnow: (0, 0, 0), (-1, 0, 0), (1, 0, 0)
   -> rho: (0, 0, 0)
   -> data2_nnow: (0, 0, 0)
-> output fields
   -> data1: (0, 0, 0)
   -> data2: (0, 0, 0)
```

Powerful tool for

- Syntax checker
- Dependency checker
- Halo-update checker
- Optimization hints (caching)
- Stage level information
- Exact access patterns
- FLOP counts
- Combine with runtime information
- Source-to-source translation

Configuration flexibility

- Developer feedback on C++ dynamical core
  - STELLA is not flexible enough for options
  - Too many compile time options
- **Note 1** Developer feedback on Fortran side is opposite → Trend to move runtime namelist options to hardcoded options (e.g. lcarlrho_advprog, ldiabf_satad, l_hor_p_grad_Mahrer, …)
- **Note 2** Compile time of compiling three (!) fast-waves solvers in current Fortran code can be significant
- **Note 3** Switches inside loop may impact performance, switches outside loop may lead to code duplication

Conditions in STELLA

- Conditional sweep execution (IJK-loop)

Code duplication
- Caches (not shown)
- Stencil stages

```
define_switch<ltadvlimiter> (

  define_case<int, 1>(
    define_sweep<cKIncrement>(
      define_caches( ........... ),
      define_stages(
        StencilStage<TrivialStage, IJRange<cIndented,0,0,0,0>, KRange<KMinimumCenter,0,0> >(),
        StencilStage<ComputeTheta0Stage, IJRange<cIndented,-1,1,-1,1>, KRange<FullDomain,0,0> >(),
        StencilStage<PPTPStage, IJRange<cComplete,0,0,0,0>, KRange<FullDomain,0,0> >(),
        StencilStage<DataStage, IJRange<cComplete,0,0,0,0>, KRange<FullDomain,0,0> >()
      )
    )
  ),

  define_case<int, 0>(
    define_sweep<cKIncrement>(
      define_caches( ........... ),
      define_stages(
        StencilStage<TrivialStage, IJRange<cIndented,0,0,0,0>, KRange<KMinimumCenter,0,0> >(),
        StencilStage<PPTPStage, IJRange<cComplete,0,0,0,0>, KRange<FullDomain,0,0> >()
        StencilStage<DataStage, IJRange<cComplete,0,0,0,0>, KRange<FullDomain,0,0> >()
      )
    )
  ),

  define_case<int, 2> (
    ...
  )

)
```

Conditions in STELLA

- **New** Conditional stage execution (IJ-loop)

```
define_sweep<cKIncrement> (
  define_caches( ..... ),

  define_stages(
    StencilStage<TrivialStage, IJRange<cIndented,0,0,0,0>, KRange<KMinim
  ),

  define_stage_switch<iadvorder>(
    define_stage_case<int, 3>(
      define_stages(
        StencilStage<HorAdv3rdOrder, IJRange<cIndented,-1,1,-1,1>, KRa
      )
    ),
    define_stage_case<int, 5>)(
      define_stages(
        StencilStage<HorAdv5thOrder, IJRange<cIndented,-1,1,-1,1>, KRange<
      )
    )
  ),

  define_stages(
    StencilStage<PPTPStage, IJRange<cComplete,0,0,0,0>, KRange<FullDomain,0,0> >()
  ),

  define_stage_if<ltadvlimiter, 1>(
    define_stages(
      StencilStage<ComputeTheta0Stage, IJRange<cIndented,-1,1,-1,1>, KRange<FullDomain,0,0> >()
    )
  )

)
```

Benefit

- Easier to introduce conditional code
- Little or no code duplication
- Both **switch/case** and **if(cond)** behaviour

Global parameters

- **NEW** Global parameter handling
- In GlobalParam_params.h name of th
  and (default / fixed) value

```
DEFINE_DYNAMIC_GLOBAL(iadv_order, 5)
DEFINE_STATIC_GLOBAL(itheta_adv, 0)
```

- Easy to switch from compile time to runtime switch
- GlbParSet will throw an error for compile time switch

- Initialization with GlbParInit()
- To set the value of a dynamic option

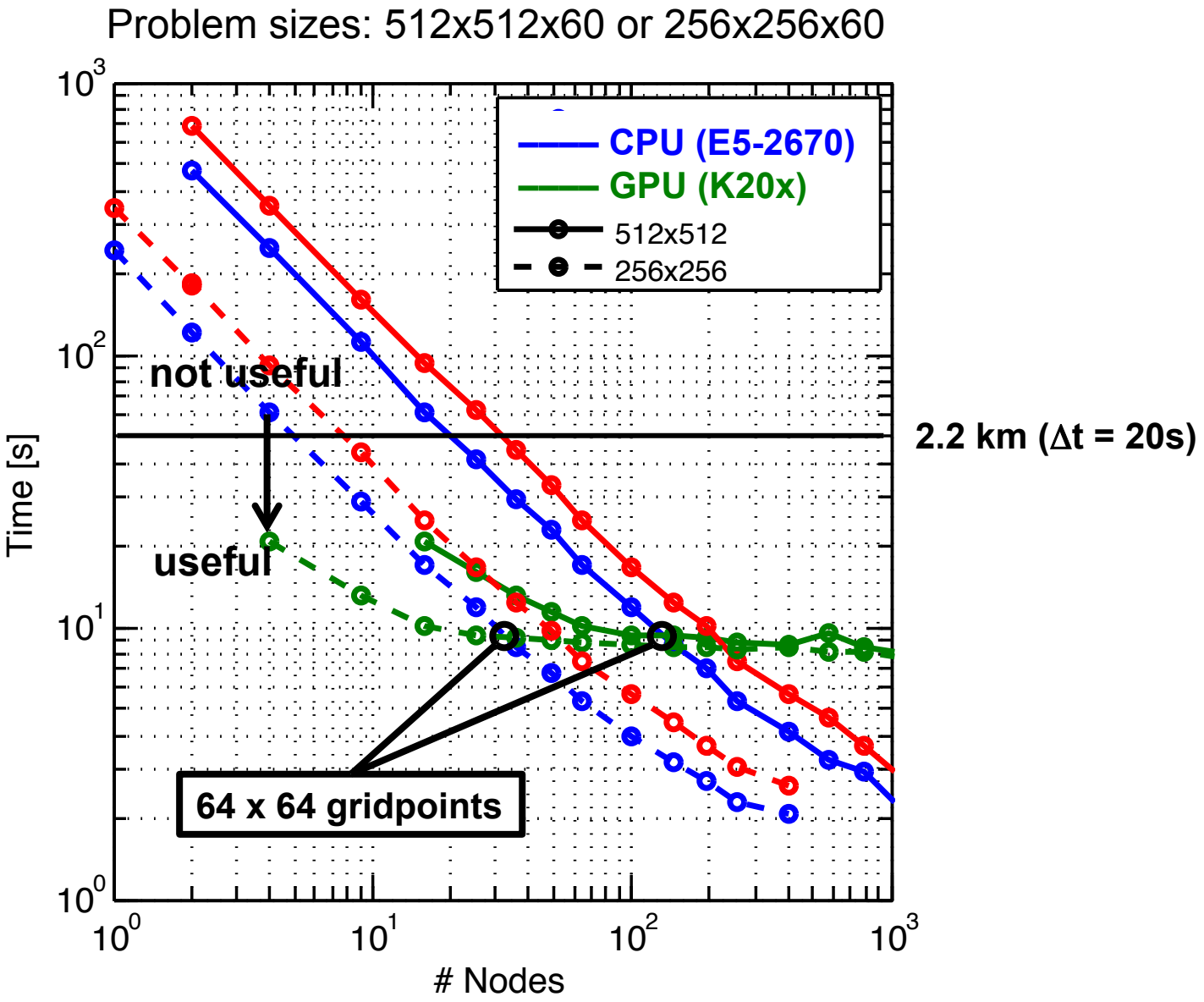
```
GlbParSet(iadv_order, 3);
GlbParSet("iadv_order", 3);
```

- To read the value of an option in device code

```
iadv = GlbParGet(iadv_order);
```

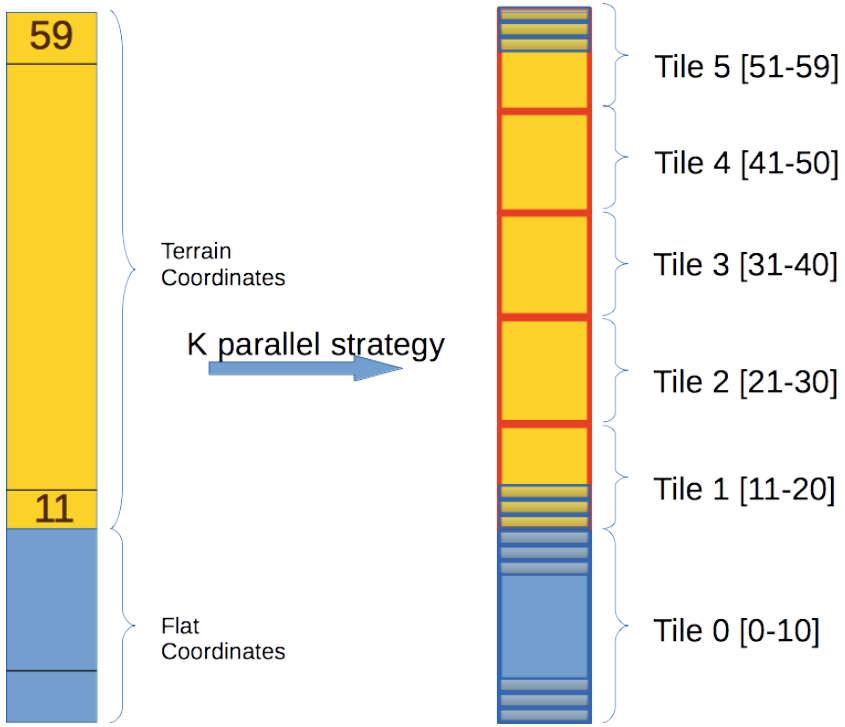
- **Important** Do not use inside loops!

Strong scalability

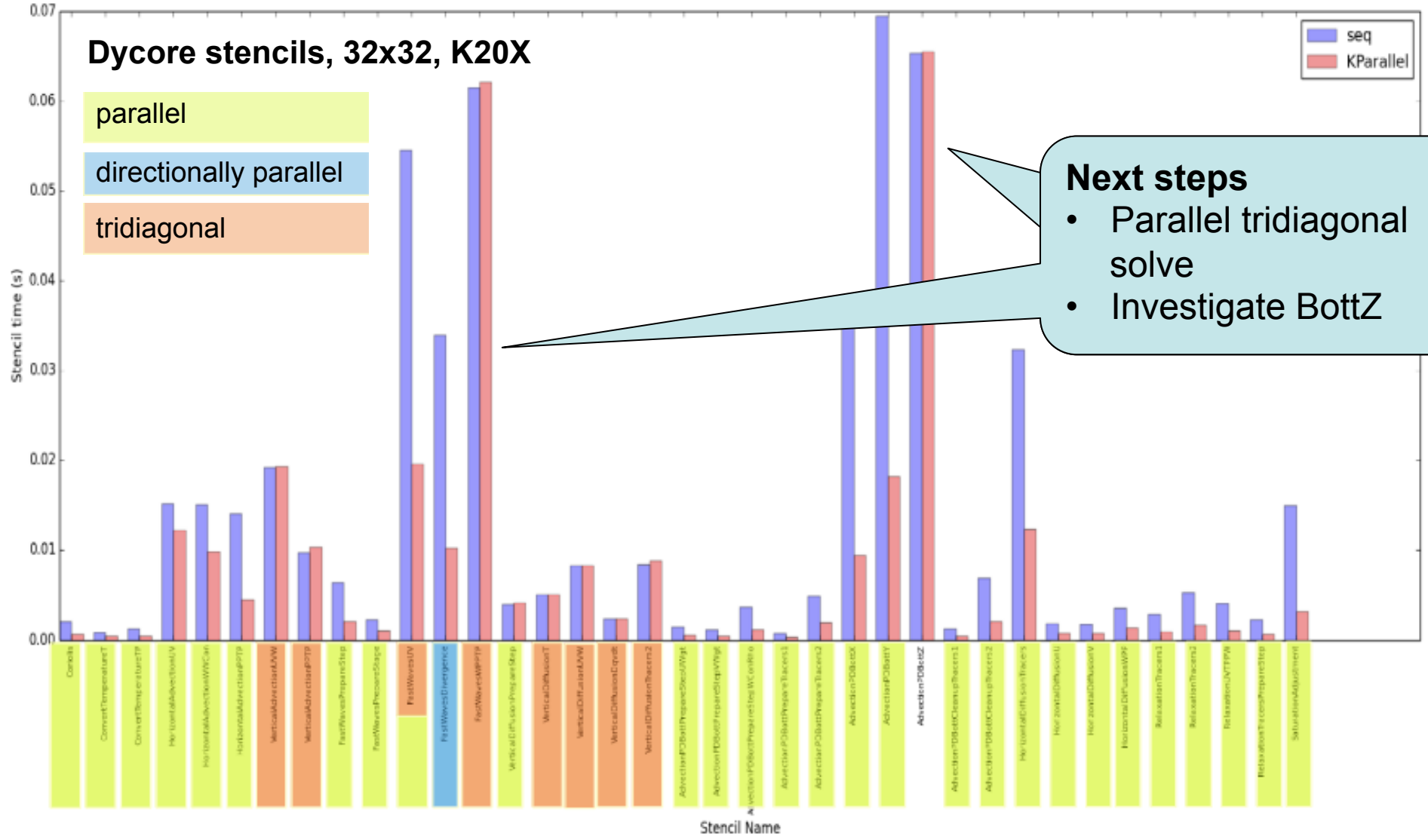


Vertical parallelization

- Parallelize along k-direction (GPU threads) in order to improve strong scalability
- API: cKIncrement, cKDecrement, **cKParallel**
- Split into blocks in order to retain ILP



Vertical parallelization

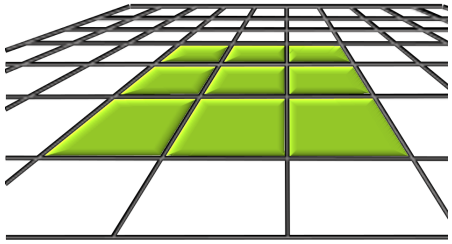


Caching

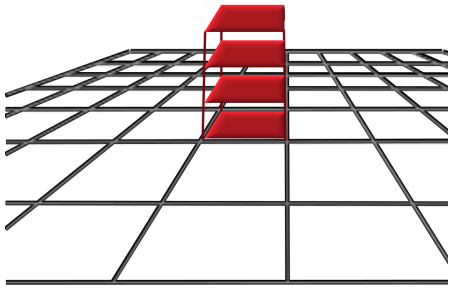
- Trend from **compute-centric** to **data-centric** programming models
- Many GPU programming models explicitly expose memory hierarchy to user
- **STELLA eases this burden** with high-level constructs (temporaries, caches, …)

NEW

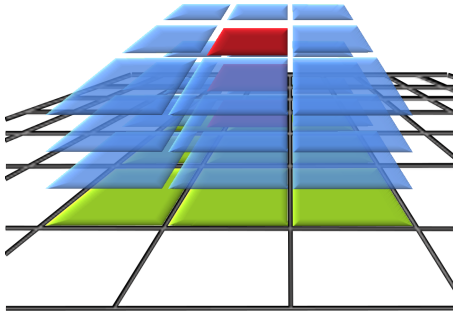
IJCache
(shared memory)



KCache
(registers)



IJKCache
(shared memory)



IJK-Caching

- Caching on 3D-accesses of temporaries

128x128	no IJKCache	IJKCache
Benchmark	33.3 ms	9.0 ms
FastWavesDivergence	0.089 ms	0.069 ms

- Caching on 3D-accesses of input fields

128x128	**no IJKCache**	**IJKCache**
5x5x3 benchmark		
time	24.4 ms	10 ms
n regs	52	34
occupancy	49%	64%
AdvectionPD Semilagrangian Interpolate (Dycore)		
time	300 ms	70 ms

- Side effect: Caching on 2D accesses of input fields

128x128	**no IJKCache (ms)**	**IJKCache (ms)**
AdvectionPDBottY	0.15	0.14
AdvectionPDBottX	0.077	0.044
HorizontalAdvectionPPTP	0.039	0.024

Python interface

- Interface

```
~:> ipython
In [1]: from stella import Dycore
In [2]: from stella.stencil import Coriolis
```

- Help

```
In [11]: Coriolis?
Type:        builtin_function_or_method
String form: <built-in function fromfile>
Docstring: Coriolis (in_u, in_v, in_fc, out_utens, out_vtens)
Applies the Coriolis stencil using the given force over the input data
fields, generating two independent output fields.
Parameters:
in_u  : input data field;
in_v  : input data field;
in_fc : a scalar representing the force applied;
...
```

- Stencil execution (NumPy compatible)

```
In [12]: d = Dycore ( )
In [13]: d._init ( (42, 27, 60) )
In [14]: v = np.random.rand (48, 33, 60)
In [15]: d.add_external_storage ('v_nnow', v)
In [16]: d._allocate_data_fields ( )
In [17]: c = Coriolis (d)
In [18]: c.do ( )
```

Python interface

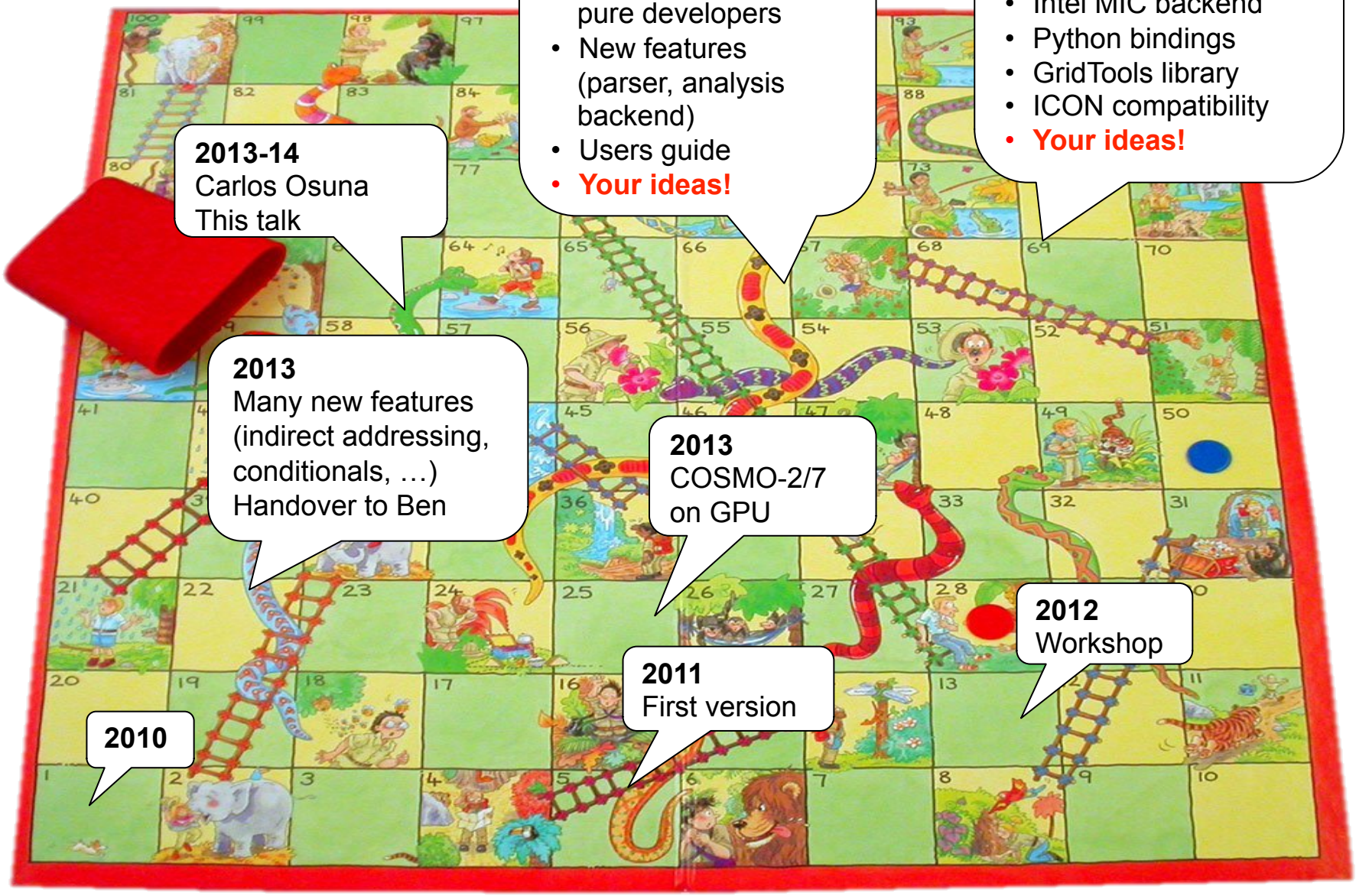
- Serialization

```
Serializer ser; ser.Init (...);
IJKRealField u; u.Init (...);
SavePoint sp; sp.Init (...);
serializer.SaveToNumPy (u, sp, "u");
```

- Next step: defining stencils (JIT)

```
# --- DEFINITION of the Coriolis stencil object
class CoriolisKernel (StencilKernel):
    """Class definition of the Coriolis stencil.-"""
    def __init__ (self, utens, vtens):
        super.__init__ (self)
        # output fields
        self.utens = utens
        self.vtens = vtens
    def _USlowTensStage (self, ctr, in_v, in_fc):
        """The 'Do' function of the U stage.-"""
        return ( in_fc * np.average ((in_v, in_v[1, 0])) +
                 in_fc * np.average ((in_v[0, -1], in_v[1, -1])) ) / 2.0
    def _VSlowTensStage (self, ctr, in_u, in_fc):
        """The 'Do' function of the V stage.-"""
        return ( in_fc * np.average (in_u[0, 0], in_u[0, 1]) +
                 in_fc * np.average (in_u[-1, 0], in_u[-1, 1]) ) / 2.0
    def kernel (self, in_u, in_v, in_fc):
        """This stencil comprises two independent stages.-"""
        for p in out_utens.interior_points (sweep='cKIncrement'):
            self.out_utens[p] += self._USlowTensStage (p, in_v, in_fc)
        for p in out_vtens.interior_points (sweep='cKIncrement'):
            self.out_vtens[p] -= self._VSlowTensStage (p, in_u, in_fc)
# --- USAGE of the Coriolis stencil object
kernel = CoriolisKernel (utens, vtens)
kernel.compilation.should_unroll = False
kernel.compilation.backend       = 'cxx'
kernel.kernel (u, v, 3.5)
```

Roadmap


2014-15
• Official version (5.2)
• Improvements for pure developers
• New features (parser, analysis backend)
• Users guide
• Your ideas!
2015-16
• Operational @ MCH
• New features
• Intel MIC backend
• Python bindings
• GridTools library
• ICON compatibility
• Your ideas!
2013-14
Carlos Osuna
This talk
2013
Many new features (indirect addressing, conditionals, …)
Handover to Ben
2013
COSMO-2/7 on GPU
2012
Workshop
2011
First version
2010

Publications

SUPERCOMPUTING FRONTIERS AND INNOVATIONS

An International Journal

TOWARDS A PERFORMANCE PORTABLE, ARCHITECTURE AGNOSTIC IMPLEMENTATION STRATEGY FOR WEATHER AND CLIMATE MODELS

Oliver Fuhrer, Carlos Osuna, Xavier Lapillonne, Tobias Gysi, Ben Cumming, Mauro Bianco, Andrea Arteaga, Thomas Christoph Schulthess

Application centric energy-efficiency study of distributed multi-core and hybrid CPU-GPU systems

Ben Cumming*, Gilles Fourestey*, Oliver Fuhrer†, Tobias Gysi‡, Massimiliano Fatica§, and Thomas C. Schulthess*¶‖

STELLA & GCL: Domain-specific tools for structured grid methods

Tobias Gysi*†, Oliver Fuhrer‡, Carlos Osuna§, Mauro Bianco¶, and Thomas C. Schulthess¶‖**

Conclusions

- Several improvements (usability, performance) in STELLA
- Improvements were all user driven
  → If you give us constructive feedback what is missing or hard, we'll work for you!
- Roadmap for the near and far future
  - STELLA will continue to improve
  - Funding secured (GridTools, Python, MIC, …)
  - More proposals pending

  → Collaboration is highly welcome (e.g. ICON)

Thank you!