Mentors

Leo Chen
NVIDIA

Min Yu
NVIDIA

NCHC Open Hackathon 2024

Final presentation

GPU-Based Acceleration VVM Team

Miles Hsieh
NTU/LCDM

Jerry Chen
NTU/AEL

Stacy Guo
NTU/AEL

Shao Tseng
NTU/AEL

Aries Fan
NTU/LCDM

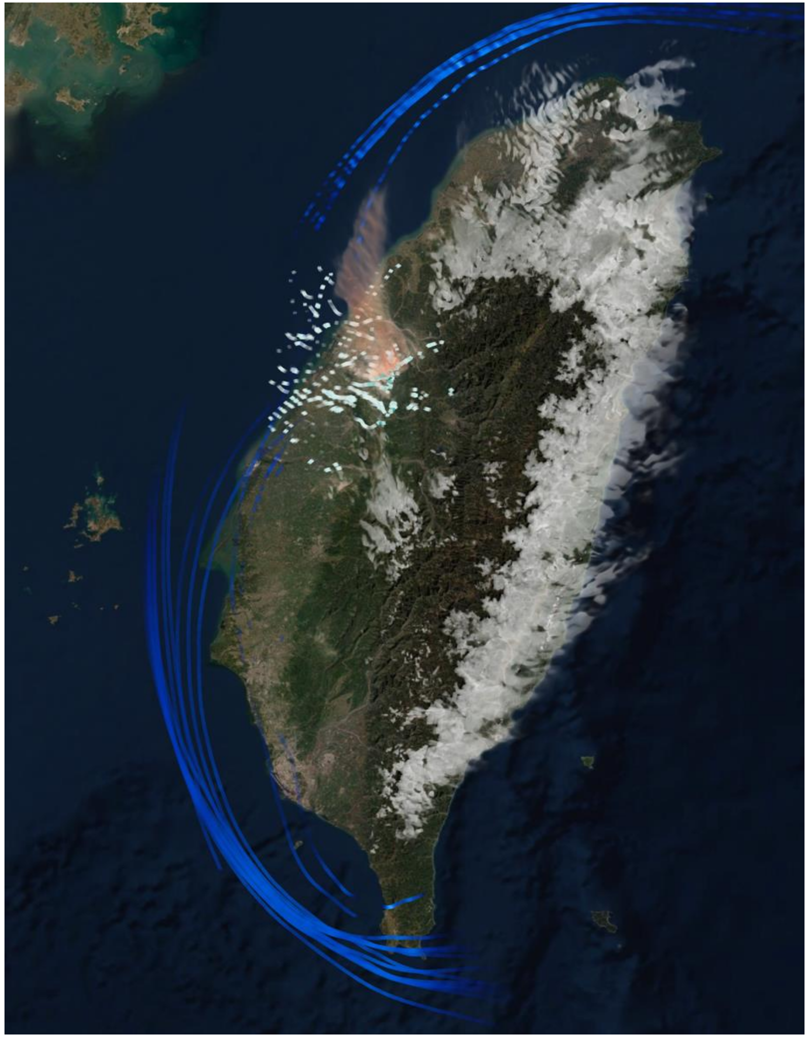
OpenACC
More Science, Less Programming

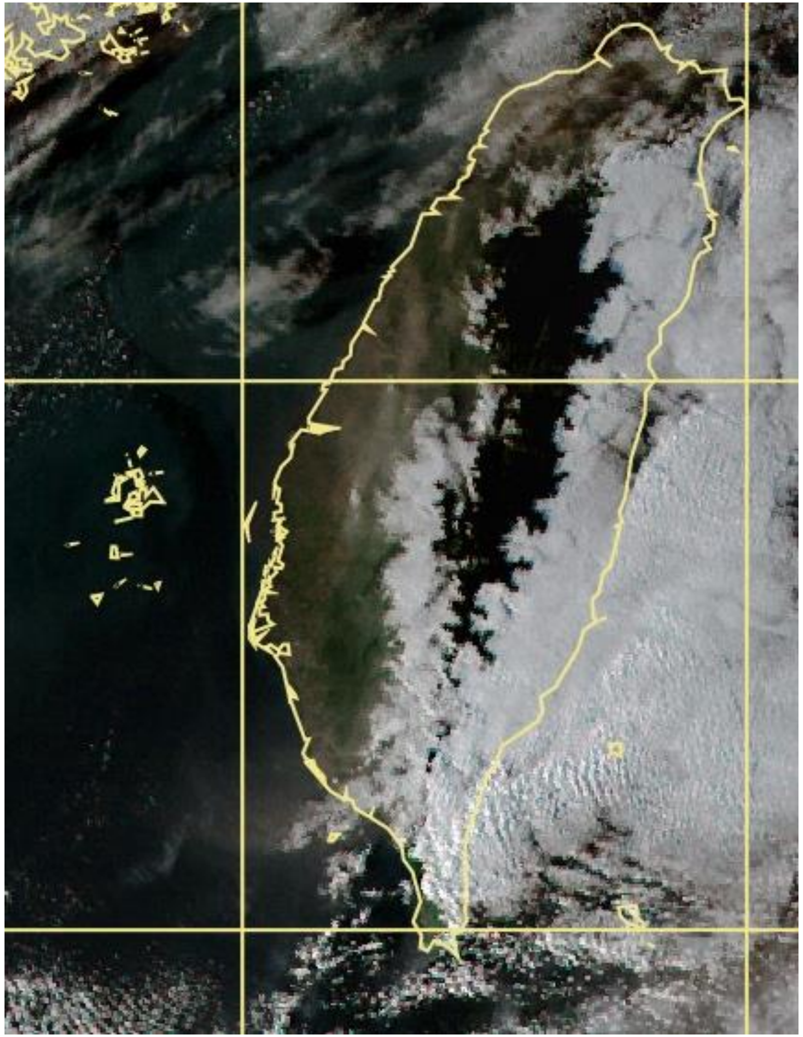
LCDM
Lab. for Cloud Dynamics and Modeling

ATMOSPHERIC ENVIRONMENT LAB

special thanks to 尚恩

Building Digital Twin of Taiwan





OpenACC More Science, Less Programming
OPEN HACKATHONS
NCHC
NARLabs 財團法人國家實驗研究院
國家高速網路與計算中心
National Center for High-performance Computing
NVIDIA

Building Digital Twin of Taiwan

OPEN HACKATHONS

NCHC

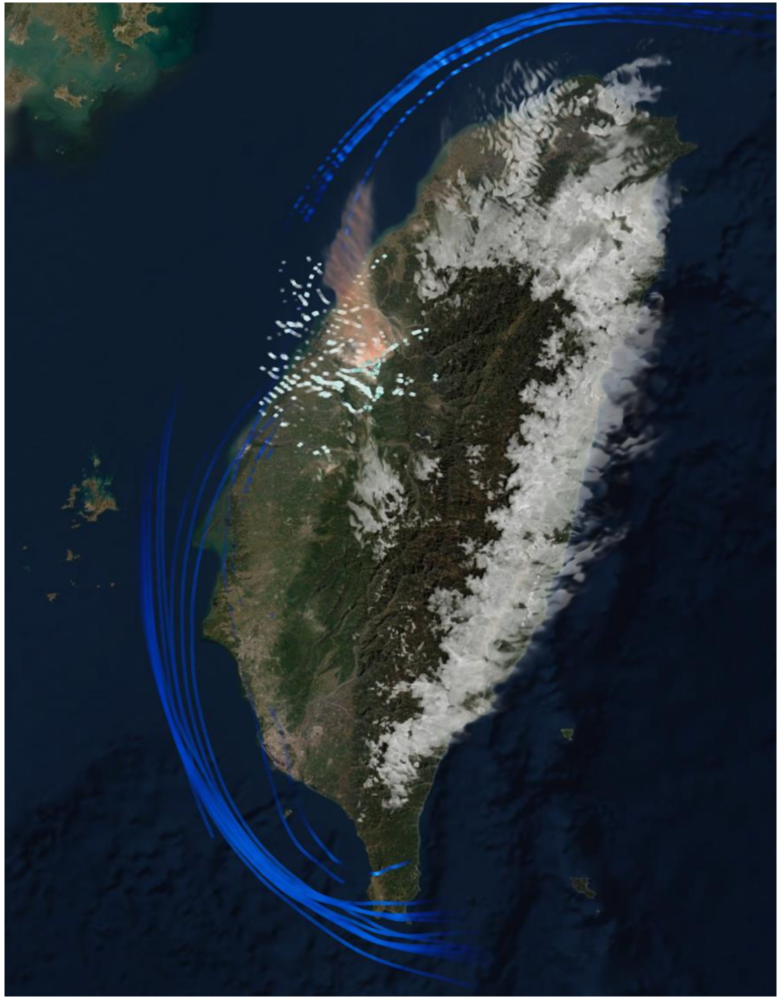
NARLabs 財團法人國家實驗研究院

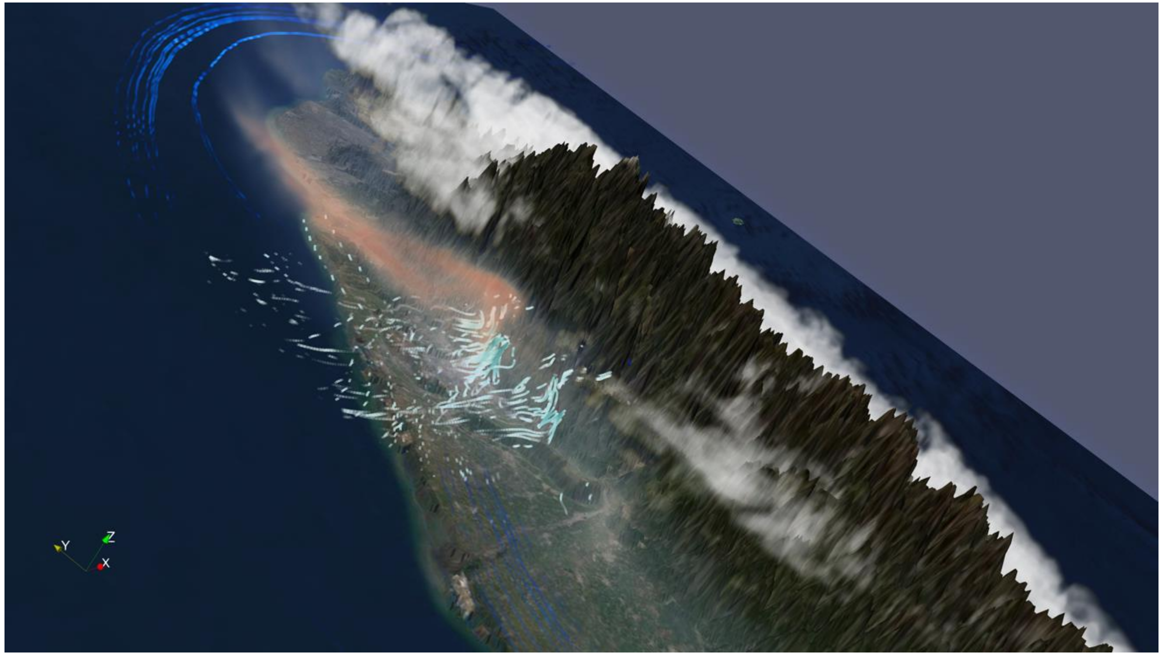
國家高速網路與計算中心

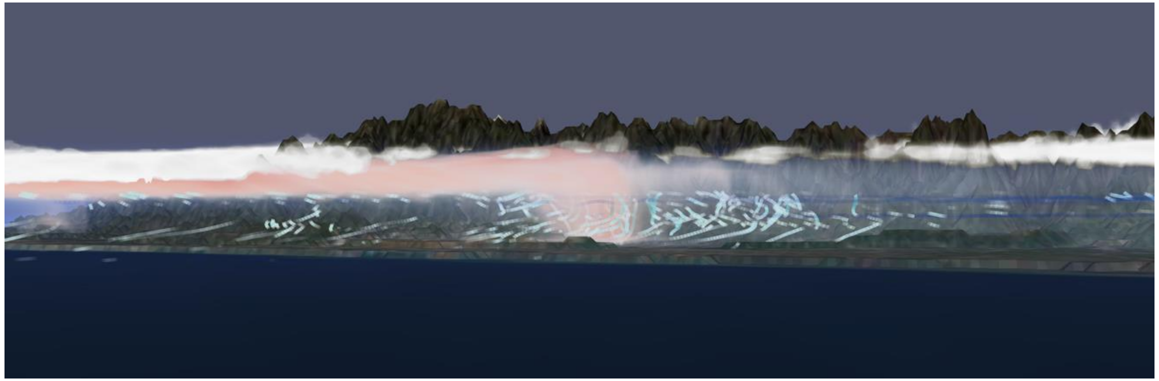
National Center for High-performance Computing

NVIDIA

Clouds, Rain, and Pollution





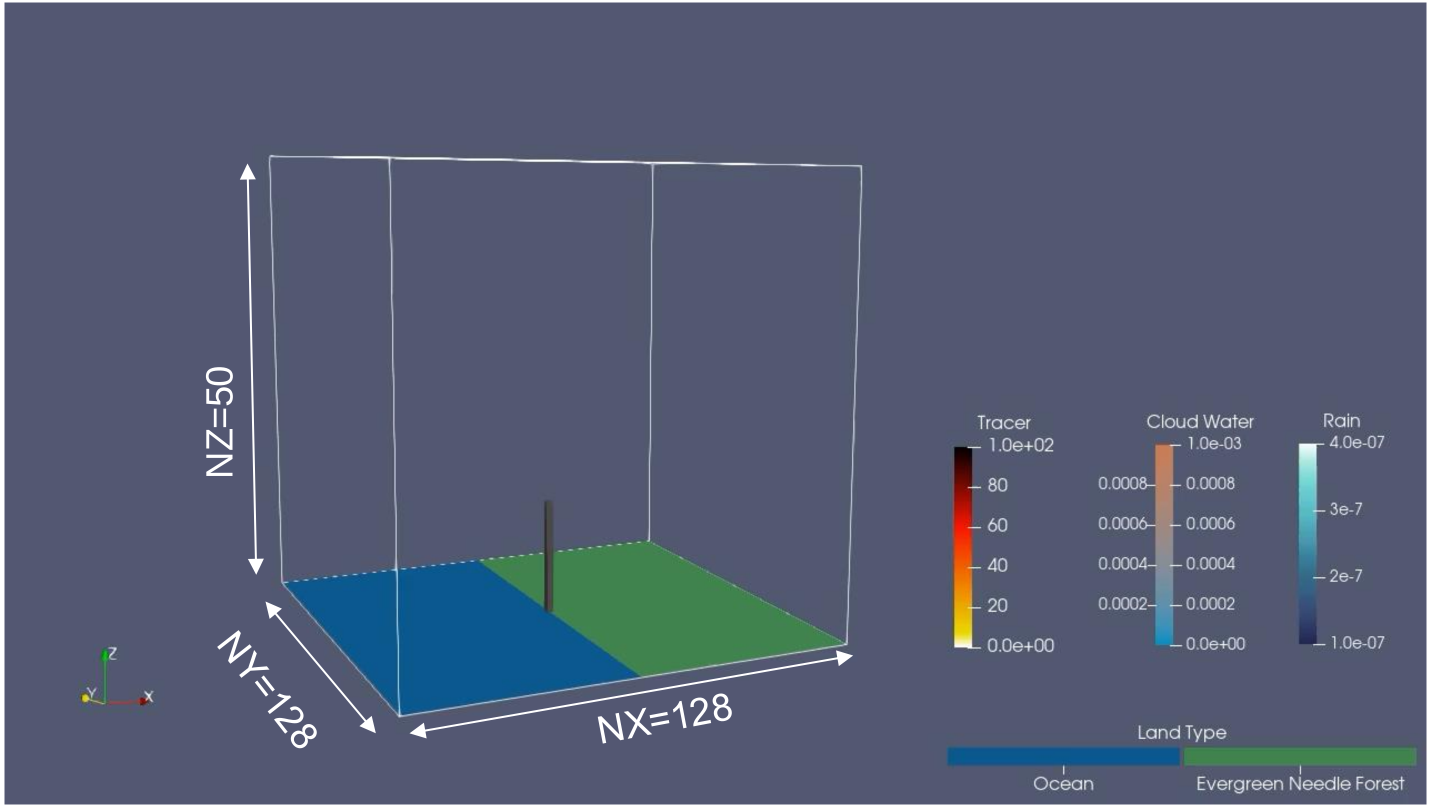


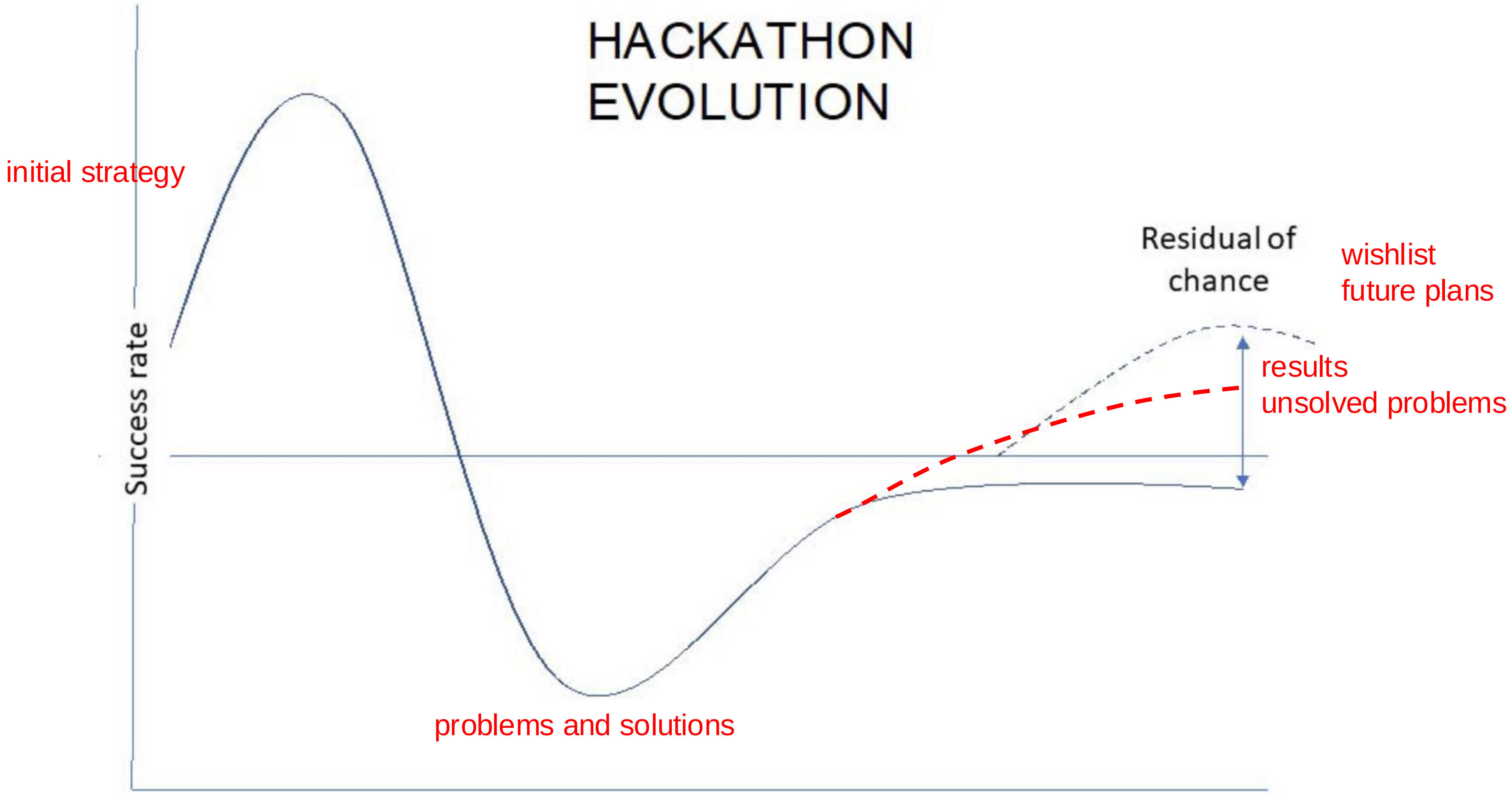
*14000 CPU hour for a 24-hr simulation

Atmospheric Physical and Chemical Processes Simulation using Vector Vorticity equation cloud-resolving Model (VVM)

- Fortran code
- 4-core CPU with MPI
- Domain: 128x128x50
- integrates 720 steps

  (2 hours)

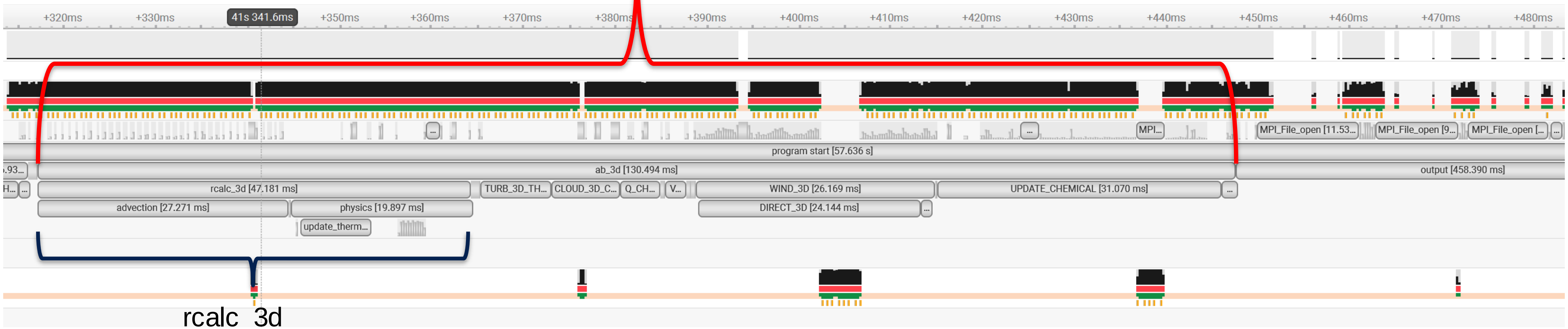




Chandrasekaran et al., (2018)
doi: 10.1109/MCSE.2018.042781332.

initial strategy

ab_3d (one step), 130.5ms



rcalc_3d

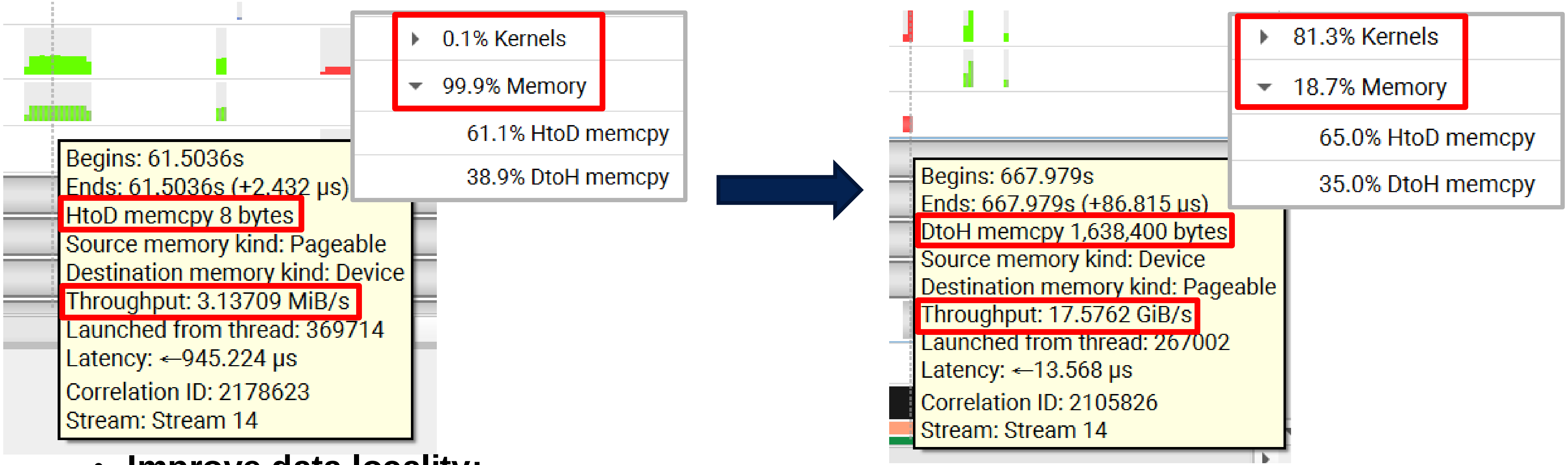
advection
20%

wind_3d
(direct_3d 18%)

UPDATE_CHEMICAL
23%

- porting path: openACC multi-core → openACC GPU

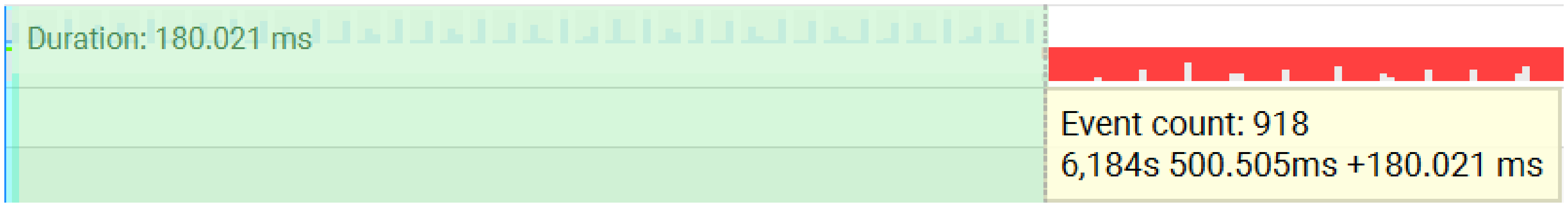
problems and solutions

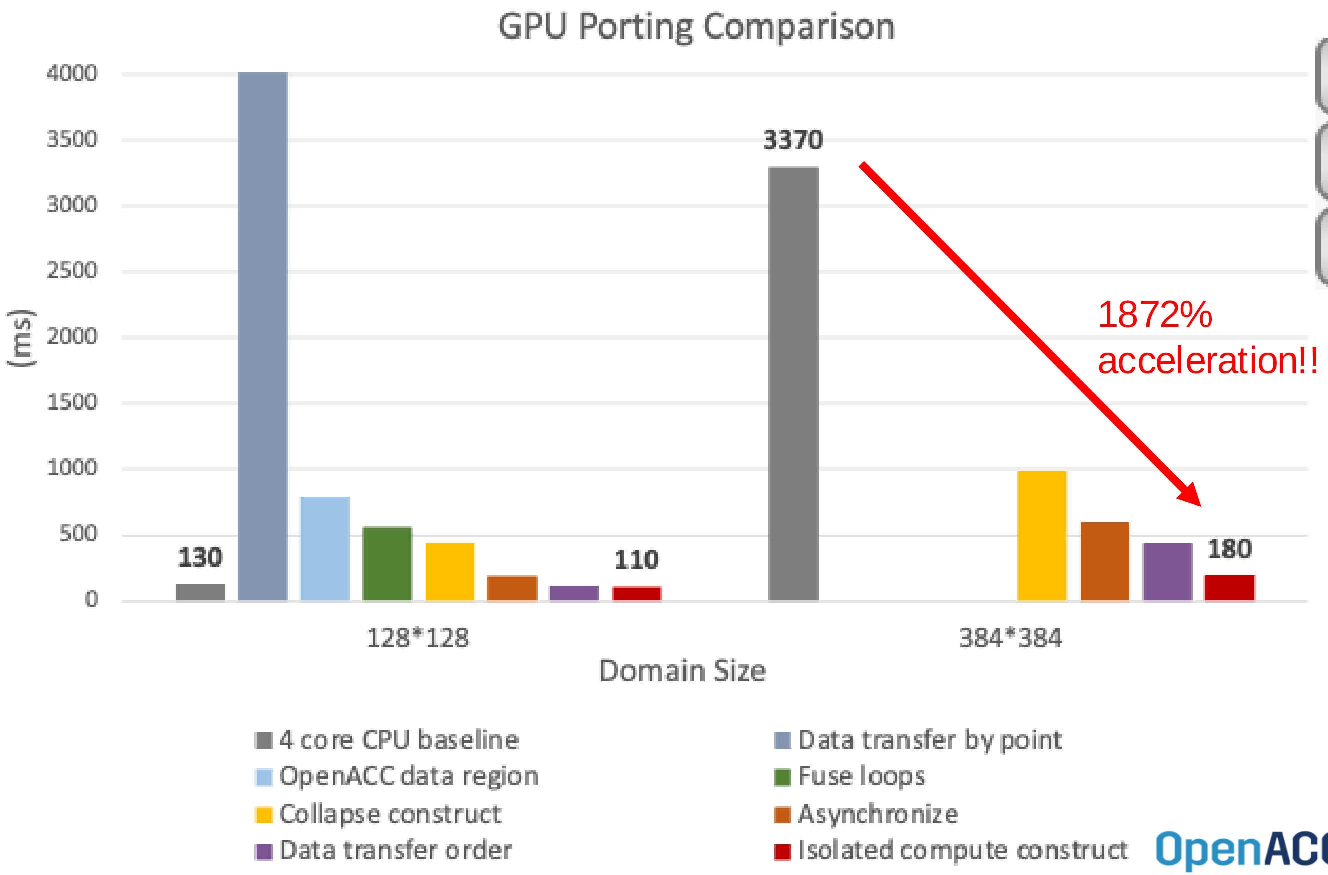


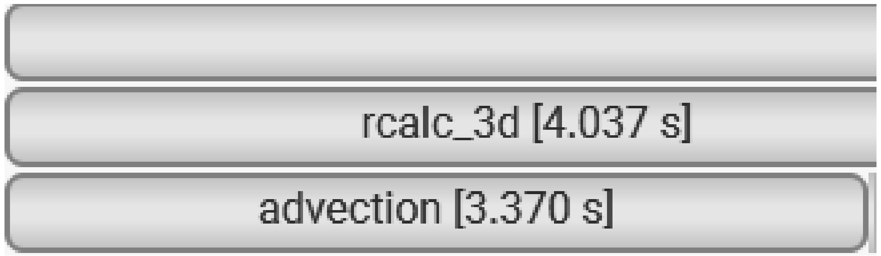
- **Improve data locality:**
- Pre-load reusable data before calling the advection module and present them in the module
- Create local variables in the GPU memory
- pruning legacy codes: reduce loops from 19 to 8
- asynchronize multiple loops

Results and Final Profile

GPU







4-core CPU

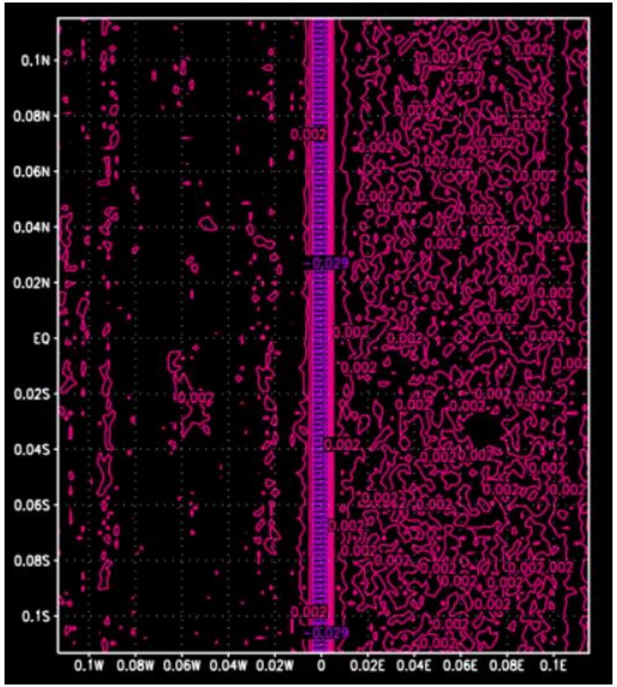
Energy Efficiency

INPUTS			
# CPU Cores	4		
# GPUs (A100)	1		
Application Speedup	18.7x		
Node Replacement	4.7x		
GPU NODE POWER SAVINGS			
	AMD Dual Rome 7742	8x A100 80GB SXM4	Power Savings
Compute Power (W)	5,143	6,500	-1,358
Networking Power (W)	217	93	124
Total Power (W)	**5,360**	**6,593**	-1,233
Node Power efficiency	**0.8x**		
ANNUAL ENERGY SAVINGS PER GPU NODE			
	AMD Dual Rome 7742	8x A100 80GB SXM4	Power Savings
Compute Power (kWh/year)	45,048	56,940	(11,892)
Networking Power (kWh/year)	1,902	814	1,088
Total Power (kWh/year)	**46,950**	**57,754**	**(10,804)**
$/kWh	0.20		
Annual Cost Savings	(2,160.71)		
3-year Cost Savings	(6,482.13)		
Metric Tons of CO_2	(8)		
Gasoline Cars Driven for 1 year	(2)		
Seedlings Trees grown for 10 years	(127)		

- **Time consumption of advection in one time step**
- **GPU acceleration is more apparent in larger-domain-computation**
- **We're able to achieve over 1860% speedup in the 384*384 domain.**
- **TaiwanVVM with 1024*1024 domain can achieve over 750% speedup in a sample test.**

unsolved problems: update_chemical module

- precision issues (not sure)




```fortran
subroutine ITER(gdt,k)
implicit none

  real    gdt
  integer k
  integer j,n,r,s  !loop count
  real :: YP_local(1:i1,1:j1), YL_local(1:i1,1:j1)

  YL_local = 0.
  YP_local = 0.
  !$acc data copyin(PL_scheme, KeffT, keff, RC, ysum, gdt, k, H2O) copy(ynew, YP, YL)
  !$acc parallel loop private(j, kreact,YP_local,YL_local)
  do n=1,nchsp  !chemical species
  if (PL_scheme(n)%active .EQV. .TRUE.) then    !reactive or not (True or False)
    if (PL_scheme(n)%name == H2O%name )  cycle    !don't do calculation of H2O
    do j=1, PL_scheme(n)%nr_PL     !chemical reactions (depend on chemical speices)
      if (RC(PL_scheme(n)%PL(j)%r_nr)%raddep == 1 ) then     !photolysis or not (1 or 0)
        select case (PL_scheme(n)%PL(j)%formula)      !select case of different formulas (8 types)
        case (0)
          if(PL_scheme(n)%PL(j)%PorL == 1) then    !production or loss (1 or 0)
              YP_local = YP_local + PL_scheme(n)%PL(j)%coef * keff(:,:,RC(PL_scheme(n)%PL(j)%r_nr)%Kindex,k)  !Production
          else
              YL_local = YL_local + PL_scheme(n)%PL(j)%coef * keff(:,:,RC(PL_scheme(n)%PL(j)%r_nr)%Kindex,k)  !Loss
          endif
        case (1~7) !other cases
        end select
      endif
    enddo   !end of chemical reactions
    do r=1,i1
      do s=1,j1
        !$acc atomic
        YP(r,s,n) = YP(r,s,n) + YP_local(r,s) !data output
        !$acc atomic
        YL(r,s,n) = YL(r,s,n) + YL_local(r,s) !data output
      enddo
    enddo
    ynew(:,:,n) = max(0.0,(ysum(:,:,n)+gdt*YP(:,:,n))/(1.0+gdt*YL(:,:,n))) !data output
  else
    ynew(:,:,n) = ysum(:,:,n)
  endif
  enddo   !end of chemical species
  !$acc end data
end subroutine ITER
```

ITT=12: `Contouring: -0 to 0.00012 interval 1e-05`

ITT=360: `Contouring: -0.016 to 0.001 interval 0.001`

ITT=720: `Contouring: -0.03 to 0.004 interval 0.001`

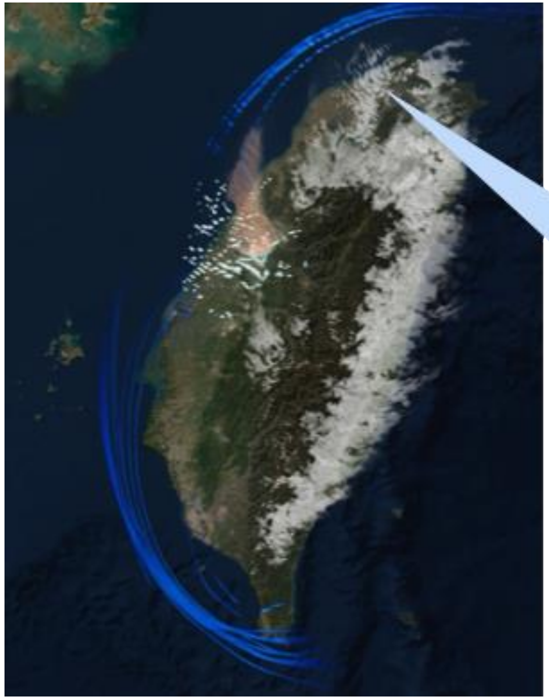
Wishlist and future plans

- keep working on update_chemical module
- focus on solving elliptical equations using CUDA
- cuSPARSE or cuFFT

$$\mu \frac{\partial w}{\partial t} + \left(\frac{\partial^2}{\partial x^2} + \frac{\partial^2}{\partial y^2}\right) w + \frac{\partial}{\partial z}\left[\frac{1}{\rho_0}\frac{\partial}{\partial z}(\rho_0 w)\right] = -\frac{\partial \eta}{\partial x} + \frac{\partial \xi}{\partial y} \qquad \text{(A.6)}$$

(Wu and Arakawa, 2011)

以GPU加速計算建構臺灣天氣的數位孿生



GBA-VVM團隊來自臺灣大學雲動力模擬暨大氣環境實驗室，將 VVM加速了18倍！！

天氣與生活息息相關，從日常是否帶傘到空汙政策規劃都需仰賴氣象資訊。臺灣地狹人稠，對劇烈天氣的時機、強度與影響範圍的掌握尤為重要。VVM是一個優異的高解析度大氣模式，能有效模擬臺灣常見的午後雷陣雨及空污天氣，但其預報模擬需大量CPU運算，受限於硬體資源有限常導致模擬時間過長，難以提供即時預報參考，且高耗能的計算亦不符合減碳趨勢。為此，將VVM運算改由GPU加速，不僅縮短模擬時間，也符合減能需求。透過程式重整與清理降低計算複雜度，並以GPU進行VVM中氣象參數及污染物的傳送計算，實現了18倍的加速效果。此成果與開發經驗有助於提升VVM其他模組效能，並將VVM轉型為以GPU為核心的高速運算氣象模式，為建構臺灣天氣數位孿生系統奠定關鍵基礎。

Optimize data management

Optimize program construct

	4 core CPU baseline	Data transfer by point	OpenACC data region	Fuse loops	Collapse construct	Asynchronize directional advection	Change data transfer order	Isolated compute construct
128*128	130	7300(--)	790(--)	560(--)	440(--)	190(--)	120(1.08x)	110(1.18x)
384*384	3370				985(3.42x)	600(5.62x)	440(7.66x)	180(18.72x)

OpenACC More Science, Less Programming · OPEN HACKATHONS · NCHC · NARLabs 財團法人國家實驗研究院 國家高速網路與計算中心 National Center for High-performance Computing · NVIDIA

Thank You

OpenACC
More Science, Less Programming