中国地质大学
China University of Geosciences

艰苦樸素 求真务實

基于HPX开发CFD模型的思考

HPX并行化方法

Async

```
hpx::future<std::uint64_t> n1 = hpx::async(fibonacci, n - 1);
hpx::future<std::uint64_t> n2 = hpx::async(fibonacci, n - 2);

return n1.get() + n2.get();   // 手动，考虑循环的依赖关系
```

Dataflow（DAG）

Component action
(Server-Client)

```
// 设置好shared_futures
shared_future<double> principal = make_ready_future(init_principal);
shared_future<double> rate = make_ready_future(init_rate);

for (int i = 0; i < t; ++i)   // 复利计算周期循环
{
    shared_future<double> interest = dataflow(unwrapping(calc), principal,rate);
    principal = dataflow(unwrapping(add), principal, interest);
}

// wait for the dataflow execution graph to be finished
double result = principal.get();   // 调用在future principle上的hpx::future::get
```

尽可能降低使用HPX对原始DSL的侵入性；

异步并行与CFD算法的适配性 （循环计算在时空上有依赖性）。

Fibonacci数列 $F(0)=0$，$F(1)=1$, $F(n)=F(n-1)+F(n-2)$（$n \geqslant 2$，$n \in N^*$）

银行复利计算： $F=P(1+i)^n$

$$U_t = K_x U_{xx} \qquad \frac{u_i^{n+1}-u_i^n}{\Delta t} = K_x \frac{u_{i+1}^n - 2u_i^n + u_{i-1}^n}{\Delta x^2}$$

多重网格法

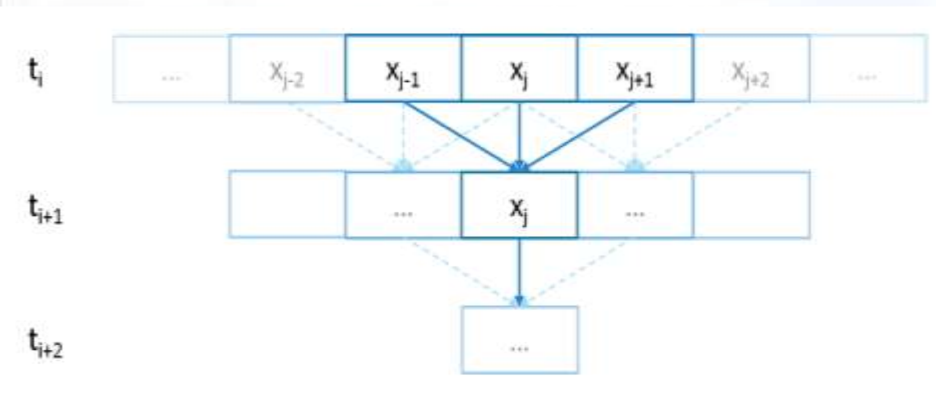
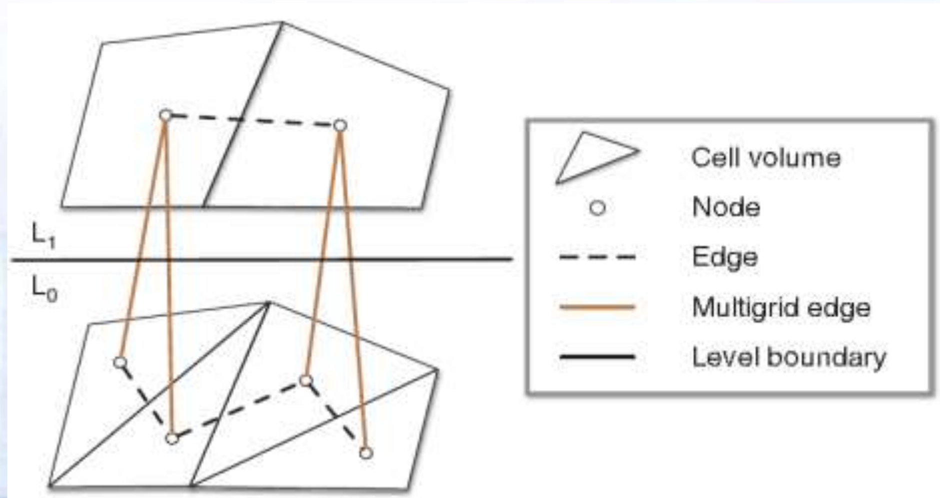
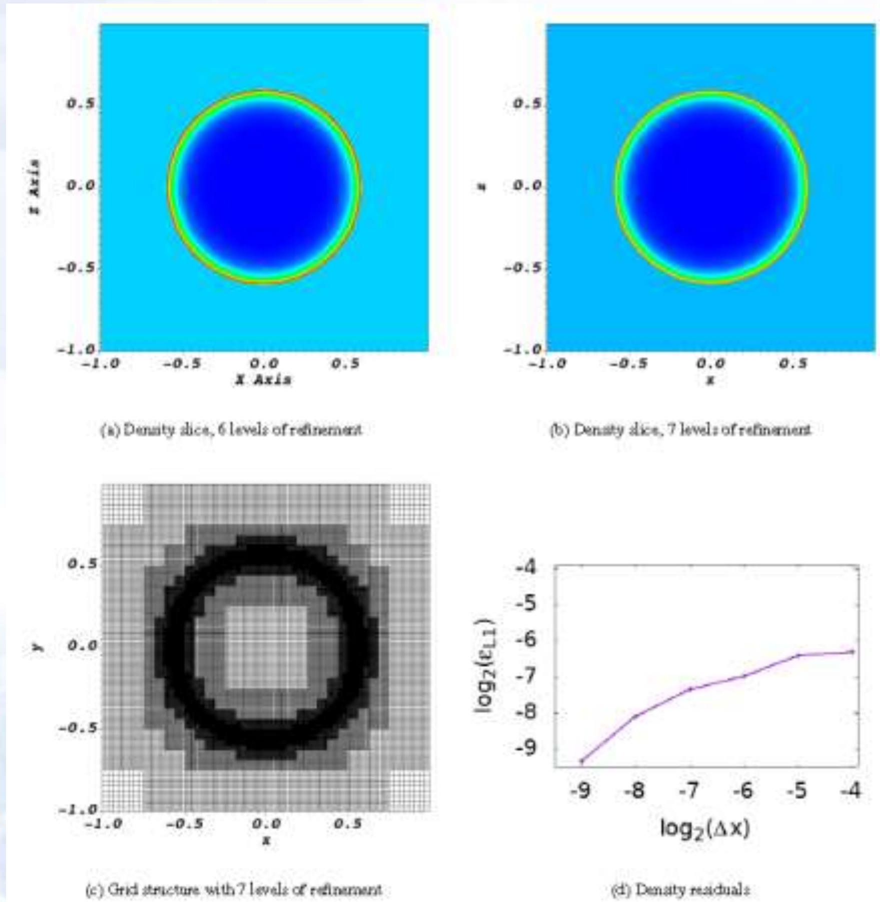


Fig. 2.2: Heat diffusion example program flow.



Octo-Tiger



(a) Density slice, 6 levels of refinement
(b) Density slice, 7 levels of refinement
(c) Grid structure with 7 levels of refinement
(d) Density residuals

Dominic C. Marcello et al. Octo-Tiger: a new, 3D hydrodynamic code for stellar mergers that uses HPX parallelisation. 2021 Mon. Not. R. Astron. Soc

基于HPX开发CFD模型的方法

(1)基于HPX (C++)的Component并行化方法，完全重新开发CFD，如DGSWE_v2 (github，HPX版本开发已停止)，nast_hpx(github)

(2)在DSL-OP2库中，实施基于单个locality的HPX并行化(Zahra, 2017);

(3)基于libgeodecomp库，实现对遗留源码(C/FORTRAN)的DGSWEM_v1改造。

HAIL-CAESAR-libgeodecomp

Zahra Khatami. Compiler and Runtime Optimization Techniques for Implementation Scalable Parallel Applications. 2017. Louisiana State University Doctoral Dissertations: https://digitalcommons.lsu.edu/gradschool_dissertations/4091

Zachary D. Byerly, Hartmut Kaiser, Steven Brus, Andreas Schaefer. A Non-intrusive Technique for Interfacing Legacy Fortran Codes with Modern C++ Runtime Systems. 2017. AMS Conference